Python Django Tutorials

吳柏翰(Jerry)

Vitrualenv

Vitrualenv

Python Virtual Envs
Django 1.5
virtualenv1
Django 1.4
virtualenv2
Django 1.0
virtualenv3
.....
Python
Easy Install
PIP
Virtualenv
Virtualenvwrapper
Local Mac
ZIP
BASH
Python 2.7
.....
Python 3.3
GIT
SQL
Homebrew
.....
Django Projects
/usr/bin
/usr/local/bin

Vitrualenv

- pip install virtualenv
- pip3 install virtualenv
- easy_install virtualenv

- virtualenv
- virtualenv project

- WIN Scripts\activate.bat
- MAC
  - source ./bin/activate
  - Deactivate

```
(m_django) c:\GitHub\mandarinx_django\m_django>
```

Install Django

pip install django==1.11rc1

```
43% |                                        | 2.9MB 12kB/s eta 0:05:2
43% |                                        | 2.9MB 13kB/s eta 0:04:5
43% |                                        | 3.0MB 16kB/s eta 0:03:5
43% |                                        | 3.0MB 14kB/s eta 0:04:1
43% |                                        | 3.0MB 16kB/s eta 0:04:0
43% |                                        | 3.0MB 13kB/s eta 0:04:5
43% |                                        | 3.0MB 14kB/s eta 0:04:2
44% |                                        | 3.0MB 13kB/s eta 0:04:4
44% |                                         | 3.0MB 14kB/s eta 0:04:
44% |                                         | 3.0MB 15kB/s eta 0:04:
44% |                                         | 3.0MB 17kB/s eta 0:03:
44% |                                         | 3.1MB 14kB/s eta 0:04:
44% |                                         | 3.1MB 13kB/s eta 0:04:
44% |                                         | 3.1MB 13kB/s eta 0:04:
45% |                                         | 3.1MB 14kB/s eta 0:04:
45% |                                         | 3.1MB 16kB/s eta 0:03:
45% |                                         | 3.1MB 14kB/s eta 0:04:
45% |                                         | 3.1MB 15kB/s eta 0:03:
45% |                                         | 3.1MB 13kB/s eta 0:04:
45% |                                         | 3.1MB 14kB/s eta 0:04:
46% |                                         | 3.1MB 13kB/s eta 0:04:
46% |                                         | 3.2MB 15kB/s eta 0:04:
46% |                                         | 3.2MB 15kB/s eta 0:03:
46% |                                         | 3.2MB 16kB/s eta 0:03:
```

Install Django - Verifying

python -m django --version

```
(mywebsite) c:\GitHub\Django_new\mysite>python -m django --version
1.11rc1
```

Part 1: Requests and responses

Creating a project

- django-admin startproject mysite
- python manage.py runserver

```
mysite/
    manage.py
    mysite/
        __init__.py
        settings.py
        urls.py
        wsgi.py
```

Creating a project

- manage.py: A command-line utility that lets you interact with this Django project in various ways.
- mysite/__init__.py: An empty file that tells Python that this directory should be considered a Python package.
- mysite/settings.py: Settings/configuration for this Django project.
- mysite/urls.py: The URL declarations for this Django project.
- mysite/wsgi.py: An entry-point for WSGI-compatible web servers to serve your project.

Creating app

- What's the difference between a project and an app?
- An app is a Web application that does something – e.g., a Weblog system, a database of public records or a simple poll app.
- A project is a collection of configuration and apps for a particular website. A project can contain multiple apps. An app can be in multiple projects.

Creating app

- python manage.py startapp jerry

```
polls/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    views.py
```

Write your first view

Write your first view

Write your first view

```
3   The `urlpatterns` list routes URLs to views. For more information please see:
4       https://docs.djangoproject.com/en/dev/topics/http/urls/
5   Examples:
6   Function views
7       1. Add an import:  from my_app import views
8       2. Add a URL to urlpatterns:  url(r'^$', views.home, name='home')
9   Class-based views
10      1. Add an import:  from other_app.views import Home
11      2. Add a URL to urlpatterns:  url(r'^$', Home.as_view(), name='home')
12  Including another URLconf
13      1. Import the include() function: from django.conf.urls import url, include
14      2. Add a URL to urlpatterns:  url(r'^blog/', include('blog.urls'))
15  """
16  from django.conf.urls import include, url
17  from django.contrib import admin
18
19  urlpatterns = [
20      url(r'^polls/', include('polls.urls')),
21      url(r'^admin/', admin.site.urls),
22  ]
23
```

Part 2: Models and the admin site

Part 2: Models and the admin site

```
$ python manage.py migrate
```

- The migrate command looks at the INSTALLED_APPS setting and creates any necessary database tables according to the database settings in your mysite/settings.py file and the database migrations shipped with the app (we'll cover those later).

Other features:

- Data validation on data Saving/Update
- Form generation with help of form
- Auto generated Admin Site for adding, deleting, editing records

Many other database with community App

- SQL Server
- MongoDB
- App Engine
- ElasticSearch
- Cassandra
- SimpleDB
- Redis

File Edit Selection Find View Goto Tools Project Preferences Help Corona Editor

OPEN FILES
- views.py
- urls.py — polls
- urls.py — mysite
- settings.py
- models.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | urls.py — polls | urls.py — mysite | settings.py | models.py

```python
from django.db import models

# Create your models here.
class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')


class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)
```

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

# Internationalization
# https://docs.djangoproject.com/en/dev/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'Asia/Taipei'

USE_I18N = True

USE_L10N = True

USE_TZ = True
```

- python manage.py makemigrations polls
- python manage.py sqlmigrate polls 0001
- python manage.py migrate

```
--
-- Create model Choice
--
CREATE TABLE "polls_choice" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "c
hoice_text" varchar(200) NOT NULL, "votes" integer NOT NULL);
--
-- Create model Question
--
CREATE TABLE "polls_question" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
"question_text" varchar(200) NOT NULL, "pub_date" datetime NOT NULL);
--
-- Add field question to choice
--
ALTER TABLE "polls_choice" RENAME TO "polls_choice__old";
CREATE TABLE "polls_choice" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "c
hoice_text" varchar(200) NOT NULL, "votes" integer NOT NULL, "question_id" integ
er NOT NULL REFERENCES "polls_question" ("id"));
INSERT INTO "polls_choice" ("question_id", "id", "choice_text", "votes") SELECT
NULL, "id", "choice_text", "votes" FROM "polls_choice__old";
DROP TABLE "polls_choice__old";
CREATE INDEX "polls_choice_question_id_c5b4b260" ON "polls_choice" ("question_id
");
COMMIT;
```

Playing with the API

- python manage.py shell
- **from** polls.models **import** Question, Choice
- Question.objects.all()
- **from** django.utils **import** timezone
- q = Question(question_text="What's new?", pub_date=timezone.now())
- q.save()
- q.id
- q.question_text
- q.pub_date

Playing with the API

- q.question_text = "What's up?“
- q.save()
- Question.objects.all()

```
>>> Question.objects.all()
<QuerySet [<Question: Question object>]>
>>>
```

Playing with the API

OPEN FILES
- views.py
- urls.py — polls
- urls.py — mysite
- settings.py
- models.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | urls.py — polls | urls.py — mysite | settings.py | models.py

```python
from django.db import models

# Create your models here.
class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')

    def __str__(self):
        return self.question_text


class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)

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

Playing with the API

- python manage.py makemigrations polls
- python manage.py sqlmigrate polls 0001
- python manage.py migrate
- python manage.py shell

- **from** polls.models **import** Question, Choice
- Question.objects.all()

```
>>> from polls.models import Question, Choice
>>> Question.objects.all()
<QuerySet [<Question: What's up>]>
```

Creating an admin user

- python manage.py createsuperuser
- Username: admin
- Email address: admin@example.com
- Password: &1234567
- python manage.py runserver

Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

Recent actions

My actions

None available

Make the poll app modifiable in the admin

Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

POLLS		
Questions	+ Add	Change

Recent actions

My actions

None available

Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Polls › Questions › Add question

Add question

Question text: What's up?

Date published: Date: 2017-03-25 Today | Time: 13:51:15 Now |

Save and add another | Save and continue editing | SAVE

Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Polls › Questions › What's up? › History

Change history: What's up?

DATE/TIME	USER	ACTION
March 25, 2017, 1:52 p.m.	admin	Added.
March 25, 2017, 1:53 p.m.	admin	No fields changed.
March 25, 2017, 1:53 p.m.	admin	No fields changed.

Part 3: Views and templates

Functions

- Question “index” page – displays the latest few questions.
- Question “detail” page – displays a question text, with no results but with a form to vote.
- Question “results” page – displays results for a particular question.
- Vote action – handles voting for a particular choice in a particular question.

polls/urls.py

OPEN FILES
- views.py
- urls.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | urls.py

```python
from django.conf.urls import url

from . import views

urlpatterns = [
    # ex: /polls/
    url(r'^$', views.index, name='index'),
    # ex: /polls/5/
    url(r'^(?P<question_id>[0-9]+)/$', views.detail, name='detail'),
    # ex: /polls/5/results/
    url(r'^(?P<question_id>[0-9]+)/results/$', views.results, name='results'),
    # ex: /polls/5/vote/
    url(r'^(?P<question_id>[0-9]+)/vote/$', views.vote, name='vote'),
]
```

polls/views.py

- http://127.0.0.1:8000/polls/34/

- http://127.0.0.1:8000/polls/34/results/

Write views that actually do something

- Each view is responsible for doing one of two things: returning an HttpResponse object containing the content for the requested page, or raising an exception such as Http404.

polls/views.py

OPEN FILES
- views.py
- urls.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - templates
        - polls
          - detail.html
          - index.html
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | urls.py

```python
from django.http import HttpResponse

# Create your views here.
from .models import Question

def index(request):
    return HttpResponse("Hello!")

def detail(request, question_id):
    return HttpResponse("You're looking at question %s." % question_id)

def results(request, question_id):
    response = "You're looking at the results of question %s."
    return HttpResponse(response % question_id)

def vote(request, question_id):
    return HttpResponse("You're voting on question %s." % question_id)

def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    output = ', '.join([q.question_text for q in latest_question_list])
    return HttpResponse(output)
```

Home › Polls › Questions

The question "五月天厲害嗎?" was added successfully.

Select question to change

ADD QUESTION +

Action: --------- Go 0 of 3 selected

QUESTION
五月天厲害嗎?
What's up?
What's up

3 questions

127.0.0.1:8000/polls/
應用程式
多个地点Ping服务器,
Quarca - Responsive
五月天厲害嗎?, What's up?, What's up

應用程式
多个地点Ping服务器,
Quarca - Responsive
投票網站
五月天厲害嗎?
What's up?
What's up

polls/templates/polls/index.html

OPEN FILES
- views.py
- index.html
- urls.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - templates
        - polls
          - detail.html
          - index.html
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | index.html | urls.py

```
{% if latest_question_list %}
    <ul>
    {% for question in latest_question_list %}
        <li><a href="/polls/{{ question.id }}/">{{ question.question_text }}</a>
        </li>
    {% endfor %}
    </ul>
{% else %}
    <p>No polls are available.</p>
{% endif %}
```

polls/views.py

OPEN FILES
- views.py
- index.html
- urls.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - templates
        - polls
          - detail.html
          - index.html
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | index.html | urls.py

```python
from django.http import HttpResponse
from django.template import loader

# Create your views here.
from .models import Question

def index(request):
    return HttpResponse("Hello!")

def detail(request, question_id):
    return HttpResponse("You're looking at question %s." % question_id)

def results(request, question_id):
    response = "You're looking at the results of question %s."
    return HttpResponse(response % question_id)

def vote(request, question_id):
    return HttpResponse("You're voting on question %s." % question_id)

def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    template = loader.get_template('polls/index.html')
    context = {
        'latest_question_list': latest_question_list,
    }
    return HttpResponse(template.render(context, request))
```

views.py | index.html | urls.py

```
<h1>投票網站</h1>
{% if latest_question_list %}
    <ul>
    {% for question in latest_question_list %}
        <li><a href="/polls/{{ question.id }}/">{{ question.
        question_text }}</a></li>
    {% endfor %}
    </ul>
{% else %}
    <p>No polls are available.</p>
{% endif %}
```

A shortcut: render()

```
21 def index(request):
22     latest_question_list = Question.objects.order_by('-pub_date')[:5]
23     template = loader.get_template('polls/index.html')
24     context = {
25         'latest_question_list': latest_question_list,
26     }
27     return HttpResponse(template.render(context, request))
28
29 def index(request):
30     latest_question_list = Question.objects.order_by('-pub_date')[:5]
31     context = {'latest_question_list': latest_question_list}
32     return render(request, 'polls/index.html', context)
```

A shortcut: render()

OPEN FILES
- views.py
- index.html
- urls.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - templates
        - polls
          - detail.html
          - index.html
      - __init__.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - db.sqlite3
    - manage.py
  - mywebsite

views.py | index.html | urls.py

```python
from django.template import loader
from django.shortcuts import render

# Create your views here.
from .models import Question

def index(request):
    return HttpResponse("Hello!")

def detail(request, question_id):
    return HttpResponse("You're looking at question %s." % question_id)

def results(request, question_id):
    response = "You're looking at the results of question %s."
    return HttpResponse(response % question_id)

def vote(request, question_id):
    return HttpResponse("You're voting on question %s." % question_id)


def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    context = {'latest_question_list': latest_question_list}
    return render(request, 'polls/index.html', context)
```

polls/templates/polls/detail.html

```
{{ question }}
```

OPEN FILES
- views.py
- index.html
- urls.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py

views.py | index.html | urls.py

```python
from django.http import HttpResponse
from django.template import loader
from django.shortcuts import render
from django.http import Http404

# Create your views here.
from .models import Question

def index(request):
    return HttpResponse("Hello!")
```

Page not found (404)

Request Method: GET
Request URL: http://127.0.0.1:8000/polls/4/
Raised by: polls.views.detail

Question does not exist

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.

- models.py
- tests.py
- urls.py
- views.py
- db.sqlite3
- manage.py
- mywebsite

```python
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    context = {'latest_question_list': latest_question_list}
    return render(request, 'polls/index.html', context)

def detail(request, question_id):
    try:
        question = Question.objects.get(pk=question_id)
    except Question.DoesNotExist:
        raise Http404("Question does not exist")
    return render(request, 'polls/detail.html', {'question': question})
```

A shortcut: get_object_or_404()

OPEN FILES
views.py

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py

views.py

```python
from django.http import HttpResponse
from django.template import loader
from django.shortcuts import get_object_or_404, render
from django.http import Http404

# Create your views here.
from .models import Question

def index(request):
```

Page not found (404)

Request Method: GET
Request URL: http://127.0.0.1:8000/polls/11/
Raised by: polls.views.detail

No Question matches the given query.

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.

- admin.py
- apps.py
- models.py
- tests.py
- urls.py
- views.py
- db.sqlite3
- manage.py
- mywebsite

```python
    return HttpResponse("You're voting on question %s." % question_id)


def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    context = {'latest_question_list': latest_question_list}
    return render(request, 'polls/index.html', context)

def detail(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    return render(request, 'polls/detail.html', {'question': question})
```

Use the template system

OPEN FILES
- views.py
- detail.html

FOLDERS
- Django_new
  - chapter1
  - mysite
    - mysite
      - __pycache__
      - __init__.py
      - settings.py
      - urls.py
      - wsgi.py
    - polls
      - __pycache__
      - migrations
      - templates
        - polls
          - detail.html
          - index.html

views.py | detail.html

```
<h1>問題{{ question.id }}</h1>
<h1>{{ question.question_text }}</h1>
<ul>
{% for choice in question.choice_set.all %}
    <li>{{ choice.choice_text }}</li>
{% endfor %}
</ul>
```

127.0.0.1:8000/polls/3/

應用程式 多个地点Ping服务器 Quarca - Responsive

問題3

五月天厲害嗎?

Removing hardcoded URLs in templates

- The problem with this hardcoded, tightly-coupled approach is that it becomes challenging to change URLs on projects with a lot of templates.

```
<li><a href="/polls/{{ question.id }}/">{{ question.question_text }}</a></li>
```

```
<li><a href="{% url 'detail' question.id %}">{{ question.question_text }}</a></li>
```

Namespacing URL names

- How does one make it so that Django knows which app view to create for a url when using the {% url %} template tag?

polls/urls.py

```
from django.conf.urls import url

from . import views

app_name = 'polls'
urlpatterns = [
    url(r'^$', views.index, name='index'),
    url(r'^(?P<question_id>[0-9]+)/$', views.detail, name='detail'),
    url(r'^(?P<question_id>[0-9]+)/results/$', views.results, name='results'),
    url(r'^(?P<question_id>[0-9]+)/vote/$', views.vote, name='vote'),
]
```

polls/templates/polls/index.html

```
<li><a href="{% url 'polls:detail' question.id %}">{{ question.question_text }}</a></li>
```

Part 4: Forms and generic views

HTML Forms

- **action** :The **action** attribute defines the action to be performed when the form is submitted.
  Normally, the form data is sent to a web page on the server when the user clicks on the submit button.

```
1  <h1>{{ question.question_text }}</h1>
2
3  {% if error_message %}<p><strong>{{ error_message }}</strong></p>{% endif %}
4
5  <form action="{% url 'polls:vote' question.id %}" method="post">
6  {% csrf_token %}
7  {% for choice in question.choice_set.all %}
8      <input type="radio" name="choice" id="choice{{ forloop.counter }}" value="{{ choice.id }}" />
9      <label for="choice{{ forloop.counter }}">{{ choice.choice_text }}</label><br />
10 {% endfor %}
11 <input type="submit" value="Vote" />
12 </form>
```

HTML Forms

- The **method** attribute specifies the HTTP method (**GET** or **POST**) to be used when submitting the form data

```
1  <h1>{{ question.question_text }}</h1>
2
3  {% if error_message %}<p><strong>{{ error_message }}</strong></p>{% endif %}
4
5  <form action="{% url 'polls:vote' question.id %}" method="post">
6  {% csrf_token %}
7  {% for choice in question.choice_set.all %}
8      <input type="radio" name="choice" id="choice{{ forloop.counter }}" value="{{ choice.id }}" />
9      <label for="choice{{ forloop.counter }}">{{ choice.choice_text }}</label><br />
10 {% endfor %}
11 <input type="submit" value="Vote" />
12 </form>
```

Using GET

http://www.somedomain.com/register.asp?name=jobe&email=jobe@electrotank.com

Using POST

http://www.somedomain.com/register.asp

HTML Forms

- **<input type="radio">** defines a **radio button**.
- Radio buttons let a user select ONLY ONE of a limited number of choices:

```
<h1>{{ question.question_text }}</h1>

{% if error_message %}<p><strong>{{ error_message }}</strong></p>{% endif %}

<form action="{% url 'polls:vote' question.id %}" method="post">
{% csrf_token %}
{% for choice in question.choice_set.all %}
    <input type="radio" name="choice" id="choice{{ forloop.counter }}" value="{{ choice.id }}" />
    <label for="choice{{ forloop.counter }}">{{ choice.choice_text }}</label><br />
{% endfor %}
<input type="submit" value="Vote" />
</form>
```

```
<!DOCTYPE html>
<html>
<body>

<form action="/action_page.php">
  <input type="radio" name="gender" value="male"> Male<br>
  <input type="radio" name="gender" value="female"> Female<br>
  <input type="radio" name="gender" value="other"> Other<br><br>
  <input type="submit">
</form>
```

Male
Female
Other

提交

```python
from django.http import HttpResponseRedirect, HttpResponse
from django.template import loader
from django.shortcuts import get_object_or_404, render
from django.http import Http404
from django.urls import reverse
```

```python
def vote(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    try:
        selected_choice = question.choice_set.get(pk=request.POST['choice'])
    except (KeyError, Choice.DoesNotExist):
        # Redisplay the question voting form.
        return render(request, 'polls/detail.html', {
            'question': question,
            'error_message': "You didn't select a choice.",
        })
    else:
        selected_choice.votes += 1
        selected_choice.save()
        # Always return an HttpResponseRedirect after successfully dealing
        # with POST data. This prevents data from being posted twice if a
        # user hits the Back button.
        return HttpResponseRedirect(reverse('polls:results', args=(question.id,)))
```

五月天厲害嗎?

You didn't select a choice.

Vote

CMS APP - Initial

First Step

- mysite
  - settings.py
  - urls.py
- cms
  - urls.py
  - views.py

mysite/settings.py

```
# Application definition

INSTALLED_APPS = [
    'cms.apps.CmsConfig'
    'polls.apps.PollsConfig',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
```

mysite/urls.py

urls.py — mysite | views.py | urls.py — cms | settings.py

```
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  url(r'^$', views.h
Class-based views
    1. Add an import:  from other_app.views import H
    2. Add a URL to urlpatterns:  url(r'^$', Home.as
Including another URLconf
    1. Import the include() function: from django.co
    2. Add a URL to urlpatterns:  url(r'^blog/', inc
"""
from django.conf.urls import include, url
from django.contrib import admin

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

cms/urls.py

```
1  from django.conf.urls import url
2
3  from . import views
4
5  urlpatterns = [
6      url(r'^$', views.index, name='index'),
7  ]
```

cms/views.py

```
1  from django.shortcuts import render
2  from django.http import HttpResponse
3  # Create your views here.
4
5  def index(request):
6      return HttpResponse("Hello, world.")
```

CMS APP - Model

- python manage.py migrate

models.py

```python
from django.db import models

# Create your models here.
class Restaurant(models.Model):
    name = models.CharField(max_length=20)
    phone_number = models.CharField(max_length=15)
    address = models.CharField(max_length=50, blank=True)


class Food(models.Model):
    name = models.CharField(max_length=20)
    price = models.DecimalField(max_digits=3, decimal_places=0)
    comment = models.CharField(max_length=50, blank=True)
    is_spicy = models.BooleanField(default=False)
    restaurant = models.ForeignKey(Restaurant)
```

Activating models

```
31  # Application definition
32
33  INSTALLED APPS = [
34      'cms.apps.CmsConfig',
35      #'polls.apps.PollsConfig',
36      'django.contrib.admin',
37      'django.contrib.auth',
38      'django.contrib.contenttypes',
39      'django.contrib.sessions',
40      'django.contrib.messages',
41      'django.contrib.staticfiles',
42  ]
```

Activating models

- python manage.py makemigrations cms
- python manage.py sqlmigrate cms 0001
- python manage.py migrate

```
--
ALTER TABLE "cms_food" RENAME TO "cms_food__old";
CREATE TABLE "cms_food" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name
 varchar(20) NOT NULL, "price" decimal NOT NULL, "comment" varchar(50) NOT NULL
 "is_spicy" bool NOT NULL, "restaurant_id" integer NOT NULL REFERENCES "cms_res
urant" ("id"));
INSERT INTO "cms_food" ("price", "name", "id", "is_spicy", "restaurant_id", "co
ment") SELECT "price", "name", "id", "is_spicy", NULL, "comment" FROM "cms_food
_old";
DROP TABLE "cms_food__old";
CREATE INDEX "cms_food_restaurant_id_0d7c2f2f" ON "cms_food" ("restaurant_id");
COMMIT;

mywebsite) c:\GitHub\Django_new\mysite>python manage.py migrate
perations to perform:
 Apply all migrations: admin, auth, cms, contenttypes, sessions
unning migrations:
 Applying cms.0001_initial... OK
```

admin.py

```
1  from django.contrib import admin
2
3  # Register your models here.
4  from .models import Restaurant, Food
5
6  admin.site.register(Food)
7  admin.site.register(Restaurant)
```

Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Cms › Foods › Add food

Add food

Name:

Price:

Comment:

Is spicy

Restaurant: ---------

Save and add another | Save and continue editing | SAVE

Select food to change

ADD FOOD +

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

NAME	PRICE
辣味義大利麵	100

1 food

settings.py | admin.py | models.py

```python
from django.db import models

# Create your models here.
class Restaurant(models.Model):
    name = models.CharField(max_length=20)
    phone_number = models.CharField(max_length=15)
    address = models.CharField(max_length=50, blank=True)

    def __str__(self):
        return self.name

class Food(models.Model):
    name = models.CharField(max_length=20)
    price = models.DecimalField(max_digits=3, decimal_places=0)
    comment = models.CharField(max_length=50, blank=True)
    is_spicy = models.BooleanField(default=False)
    restaurant = models.ForeignKey(Restaurant)

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

Home › Cms › Foods › 辣味義大利麵

Change food

HISTORY

Name: 辣味義大利麵

Price: 100

Comment: 特別美食

☑ Is spicy

Restaurant: 意素家

Delete | Save and add another | Save and continue editing | SAVE

CMS APP - Shell

Display

- python manage.py shell
- from cms.models import Restaurant, Food
- restaurants=Restaurant.objects.all()
- restaurants

Create

- from cms.models import Restaurant, Food
- r1 = Restaurant.objects.create(name='台科小炒', phone_number ='02-88888', address='台北市基隆路')
- r1
- Restaurant.objects.all()

```
>>> Restaurant.objects.all()
<QuerySet [<Restaurant: 意素家>, <Restaurant: 中文小館>, <Restaurant: 台科小炒>
>
```

Update

- from cms.models import Restaurant, Food
- Food.objects.all()[0].price
- Food.objects.filter(name='辣味義大利麵').update(price=200)

```
>>> Food.objects.all()[0].price
Decimal('100')
```

```
>>> Food.objects.all()[0].price
Decimal('200')
>>>
```

Delete

- from cms.models import Restaurant, Food
- Restaurant.objects.all()

- f = Restaurant.objects.get(name='台科小炒')
- f.delete()

```
<QuerySet [<Restaurant: 意素家>, <Restaurant: 中文小館>, <Restaurant: 台科小炒>]
>
>>>
```

```
<QuerySet [<Restaurant: 意素家>, <Restaurant: 中文小館>]>
>>>
```

CMS APP - Views

```python
from django.shortcuts import render_to_response
# Create your views here.
from .models import Restaurant, Food

def index(request):
    restaurants = Restaurant.objects.all()
    return render_to_response('cms/menu.html',locals())
```

```
<h1 class="text-center">美食菜單</h1>
<div class="container-fluid">
{% for r in restaurants %}
<h2>{{ r.name }}</h2>
{% if r.food_set.all %}
<p>本餐廳共有{{ r.food_set.all|length }}道菜</p>
<table class="table">
    <tr>
        <th>菜名</th>
        <th>價格</th>
        <th>辣不辣</th>
        <th>註解</th>
    </tr>
    {% for food in r.food_set.all %}
    <tr>
        <td>{{ food.name }}</td>
        <td>{{ food.price }}</td>
        <td>{% if food.is_spicy %} 辣 {% else %} 不辣 {% endif %}</td>
        <td>{{ food.comment }}</td>
    </tr>
    {% endfor %}
</table>
    {% else %}
    <p>本餐廳啥都沒賣</p>
    {% endif %}
    {% endfor %}
```

美食菜單

意素家

本餐廳共有1道菜

菜名	價格	辣不辣	註解
辣味義大利麵	200	辣	特別美食

中文小館

本餐廳啥都沒賣