**中文文档：**https://yiyibooks.cn/information/xx/Django\_1.11.6/

**django项目博客：**http://www.conyli.cc/

**django资料下载：**http://down.51cto.com/search.php?q=django

1. Django简介

Django是Python写的一个开源web框架，需要部署到Web服务器使用。

**1、常见的web框架**

django：功能强大，组件齐全

tornado：小型web框架

flask：微型

**2、MTV模型(Django工作原理)**

Web服务器

URLConfig路由系统

View视图(V)

Modules模型(M)

Templates模板(T)

- 客户端通过URL访问web服务器

- Django接收并在URLConfig路由系统(urls.py)查找对应的视图函数(views.py)来处理http请求

- 视图函数(View)调用相应的数据模型(modules.py)来存取数据

- 视图函数(View)调用相应的模板(templates/\*.html)向用户展示页面

- 视图函数处理结束后，返回一个http的响应给web服务器

- web服务器将相应返回给客户端

二、Django应用基础

**1、py虚拟环境安装django**

1）新建虚拟环境

[root@room9pc01 ~]# python3 -m venv /opt/djenv/

2）激活虚拟环境

[root@room9pc01 ~]# source /opt/djenv/bin/activate

(djenv) [root@room9pc01 ~]# which python

/opt/djenv/bin/python

3）在虚拟环境下安装django

(djenv) [root@room9pc01 ~]# cd 下载/zzg\_pypkgs/dj\_pkgs/

(djenv) [root@room9pc01 dj\_pkgs]# pip install \*

1. 查看django

(djenv) [root@room9pc01 dj\_pkgs]# python

>>> import django

>>> django.\_\_version\_\_

'1.11.6'

1. 安装其他软件

(djenv) [root@room9pc01 zzg\_pypkgs]# ls /root/下载/zzg\_pypkgs

ansible-cmdb\_pkgs ansible\_pkg pymysql\_pkgs sqlalchemy\_pkgs

(djenv) [root@room9pc01 zzg\_pypkgs]# cd ansible\_pkg/

(djenv) [root@room9pc01 ansible\_pkg]# pip install \*

... ...

**2、django管理项目**

1）创建名为mysite的项目

(djenv) [root@room9pc01 day18]# django-admin startproject mysite

#在当前目录下创建mysite项目

(djenv) [root@room9pc01 day18]# cd mysite/

(djenv) [root@room9pc01 mysite]# tree .

.

├── manage.py #命令行工具，用于项目管理

└── mysite #项目配置目录

├── \_\_init\_\_.py #py2必须的文件，将目录变成包

├── settings.py #项目配置文件

├── urls.py #路由系统文件，程序入口

└── wsgi.py #用于发布项目

1. 测试开发服务器，查看页面

(djenv) [root@room9pc01 mysite]# python manage.py runserver

[root@room9pc01 ~]# firefox 127.0.0.1:8000

#默认为8000端口

1. 修改django配置

(djenv) [root@room9pc01 mysite]# vim mysite/settings.py

ALLOWED\_HOSTS = '\*' #ACL访问控制，允许所有

MIDDLEWARE = [

... ...

# 'django.middleware.csrf.CsrfViewMiddleware', #关闭安全功能

... ...

]

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

#绑定django提供的mysql引擎

'NAME': 'dj1808', #数据库名，在步骤4)创建

'USER': 'root',

'PASSWORD': '123',

'HOST': '127.0.0.1',

'PORT': '3306',

}

}

LANGUAGE\_CODE = 'zh-hans' #中文

TIME\_ZONE = 'Asia/Shanghai' #时区

USE\_TZ = False #不使用默认时区

1. 创建mysql数据库

[root@room9pc01 ~]# mysql -uroot -p123

MariaDB [(none)]> CREATE DATABASE dj1808 DEFAULT CHARSET utf8;

1. 尝试启动测试服务器

报错：MySQLdb module: No module named 'MySQLdb'。是因为django连接mysql数据库时，使用的模块是MySQLdb，py3不支持。需要作以下改动：

(djenv) [root@room9pc01 mysite]# vim mysite/\_\_init\_\_.py

import pymysql

pymysql.install\_as\_MySQLdb()

6）启动测试服务器，运行在\*:80端口

(djenv) [root@room9pc01 mysite]# python manage.py runserver 0:80

[root@room9pc01 ~]# firefox 127.0.0.1 #查看前台页面

1. 访问后台管理页面

[root@room9pc01 ~]# firefox 127.0.0.1/admin

1. 生成数据库的表

(djenv) [root@room9pc01 mysite]# python manage.py makemigrations

(djenv) [root@room9pc01 mysite]# python manage.py migrate

9）创建超级用户

(djenv) [root@room9pc01 mysite]# python manage.py createsuperuser

Username (leave blank to use 'root'):

Email address:

Password:

Password (again):

1. 配置pycharm使用新的解释器

File -> Settings -> Projcet：gjq\_day18 -> Project Interpreter ->

右上角齿轮.add local -> 勾选Existing env选择/opt/djenv/bin/python

File -> Settings -> Languages & Frameworks -> Django ->

勾选 Enable Django Support ->

Django projcet root：/root/PycharmProjects/gjq\_day18/mysite ->

Settings：mysite/settings.py

**3、django管理应用**

项目是特定网站的配置和n个应用的集合；一个应用编写完成后，可以安装到任何项目（直接拷贝目录）；django自带一些系统应用。

1. 创建名为polls的应用

(djenv) [root@room9pc01 mysite]# python manage.py startapp polls

(djenv) [root@room9pc01 mysite]# ls

db.sqlite3 manage.py mysite polls

1. 将应用集成到项目中

(djenv) [root@room9pc01 mysite]# vim mysite/settings.py

INSTALLED\_APPS = [

... ...

'polls',

]

1. 规划URL

http://x.x.x.x/polls/ #投票首页，显示所有投票项目

http://x.x.x.x/polls/1/ #显示投票详情

http://x.x.x.x/polls/1/result/ #显示投票结果

http://x.x.x.x/polls/1/vote/ #用于接收投票时提交的表单数据

... ...

4）url授权（将polls应用的网址，交给polls应用处理）

(djenv) [root@room9pc01 mysite]# vim mysite/urls.py

from django.conf.urls import url, include

from django.contrib import admin

urlpatterns = [

url(r'^admin/', admin.site.urls),

url(r'^polls/', include('polls.urls')),

]

(djenv) [root@room9pc01 mysite]# vim polls/urls.py

from django.conf.urls import url

urlpatterns = [

]

1. 为url创建路由映射

(djenv) [root@room9pc01 mysite]# vim polls/urls.py

from django.conf.urls import url

from . import views #相对导入

urlpatterns = [

#格式:url('正则匹配的url', 视图函数, 给url定义别名)

url(r'^$', views.index, name='index')

#此处'^$'，对应http://x.x.x.x/polls/

]

1. 为投票首页编写视图函数

from django.shortcuts import render, HttpResponse

def index(request):

#形参request随便定义，客户端请求将作为实参导入

return HttpResponse('<h1>投票首页</h1>')

1. 测试投票首页

(djenv) [root@room9pc01 mysite]# python manage.py runserver 0:80

[root@room9pc01 ~]# firefox 127.0.0.1/polls/

1. 修改index视图函数，使用模板

(djenv) [root@room9pc01 mysite]# mkdir polls/templates

#创建polls应用的模板目录

(djenv) [root@room9pc01 mysite]# vim polls/views.py

from django.shortcuts import render

def index(request):

return render(request, 'index.html')

#修改视图函数，使用模板返回

(djenv) [root@room9pc01 mysite]# vim polls/templates/index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>投票首页</title>

</head>

<body>

<h1>这是投票首页</h1>

</body>

</html>

三、Django模型

**1、ORM映射**

模型(表)：每个模型都用一个类表示，该类继承自django.db.models.Model

模型的类变量(字段)：在模型中每个类变量代表数据库中的一个字段

1）表的规划

计划构建2张表，一张表用于存储问题，另一张表用于存储选项

question表：id、问题内容、pub\_date

choices表：id、选项内容、选项票数、question.id

1. 编写问题模型

(djenv) [root@room9pc01 mysite]# vim polls/models.py

from django.db import models

class Question(models.Model): #继承父类

question\_text = models.CharField(max\_length=200, null=False, unique=True)

pub\_date = models.DateField()

def \_\_str\_\_(self):

return '<问题：%s>' % self.question\_text

3）编写选项模型

(djenv) [root@room9pc01 mysite]# vim polls/models.py

... ...

class Choice(models.Model):

choice\_text = models.CharField(max\_length=200)

votes = models.IntegerField(default=0)

q = models.ForeignKey(Question, on\_delete=models.CASCADE)

#使用Question的主键作外键，同步删除

def \_\_str\_\_(self):

return '<%s=>%s>' % (self.question, self.choice\_text)

**2、管理数据库**

4）基于模型生成数据表

(djenv) [root@room9pc01 mysite]# python manage.py makemigrations

(djenv) [root@room9pc01 mysite]# python manage.py migrate

#创建表：polls\_question、polls\_choice

#表名：应用名\_类名

#字段名：类变量。如果无设置主键，默认创建名为id的主键

#模型中定义了一个名叫q的外键，选项表自动出现一个名为q\_id的字段作外键

5）修改选项表的外键名称，将q改为question

(djenv) [root@room9pc01 mysite]# vim polls/models.py

class Choice(models.Model):

... ...

question = models.ForeignKey(Question, ... ...)

... ...

(djenv) [root@room9pc01 mysite]# python manage.py makemigrations

(djenv) [root@room9pc01 mysite]# python manage.py migrate

#选项表的外键名变为question\_id

6）将模型注册到后台

(djenv) [root@room9pc01 mysite]# vim polls/admin.py

from django.contrib import admin

from .models import Question, Choice

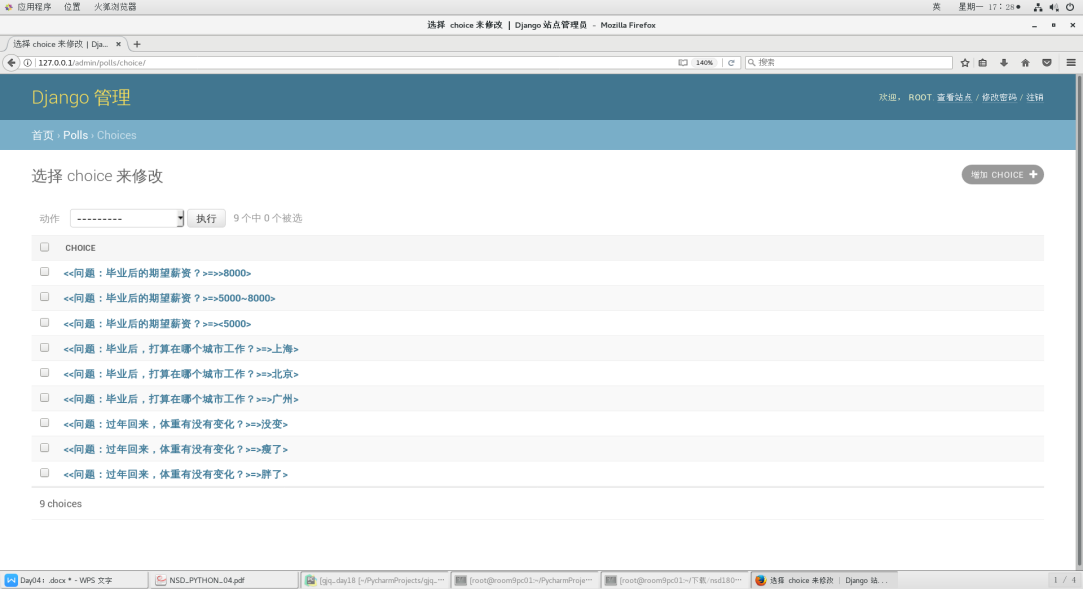
admin.site.register(Question)

admin.site.register(Choice)

7）进入后台管理界面，添加一些问题和选项

(djenv) [root@room9pc01 mysite]# python manage.py runserver 0:80

[root@room9pc01 ~]# firefox http://127.0.0.1/admin/



#在管理界面上显示的字符，实质上是\_\_str\_\_()的返回值