目录

[Django 编程基础 1](#_Toc476317778)

[Model 1](#_Toc476317779)

[model filter 查找数据时使用正则 1](#_Toc476317780)

# Django 编程基础

## 临时使用

int(x, base=10)

默认是10进制，如果想计算一个2进制的数等于10进制的多少时，可以用base=2

int(“110”, base=2)

指定二进制数为110,计算出十进制为6

### functools

#### partial

修改函数签名，使用一些参数固化，以生成一个更为简单的函数

示例：

1. 生成接受一个二进制字符串参数的函数，计算出十进制的值

>>>basetwo = partial(int, base=2)

>>>basetwo("11111") #31

#### wraps

作用：对原函数的\_\_doc\_\_、\_\_name\_\_、\_\_module\_\_、\_\_dict\_\_ 重新设定

源码：

def wraps(wrapped,

assigned = WRAPPER\_ASSIGNMENTS,

updated = WRAPPER\_UPDATES):

return partial(update\_wrapper, wrapped=wrapped,

assigned=assigned, updated=updated)

示例：

from functools import wraps

def my\_decorator(f):

@wraps(f)

def wrapper(\*args, \*\*kwargs):

print "Calling decorated function"

return f(\*args, \*\*kwargs)

return wrapper

@my\_decorator

def foo():

"""Docstring"""

print 'Called foo function'

foo()

print foo.\_\_name\_\_

print foo.\_\_doc\_\_

如果不在装饰器上添加wraps,则取不到原函数的\_\_doc\_\_, \_\_name\_\_也变为修饰器的wrapper

## tutorial

1. 查看django版本

python -m django --version

1. 创建项目

django-admin startproject mysite

项目结构如下：

mysite/

manage.py

mysite/

\_\_init\_\_.py

settings.py

urls.py

wsgi.py

1).最外层的mysite/根目录只是用来包含项目，名字并不重要，可以随时重命名

2).manage.p 命令行工具，可以让你以交互的方式与你的项目沟通

3).里面的mysite/ 目录是实际的Python package

1. 运行开发环境服务

python manage.py runserver

1. url() 函数用法

语法：url(regex, view, kwargs=None, name=None, prefix='')

1. regex 正则表达式
2. view 与regex匹配的view
3. kwargs
4. name 在模板里可以用到这个变量
5. 数据库配置
   1. 默认是配置sqlite
   2. 最常用的数据库引擎

'django.db.backends.sqlite3',

'django.db.backends.postgresql',

'django.db.backends.mysql',

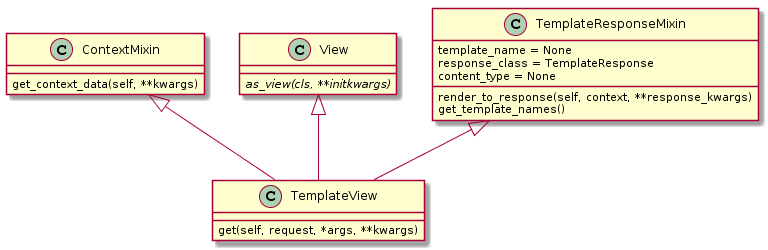
'django.db.backends.oracle'

1. 创建超级管理员

python manage.py createsuperuser

## view

### TemplateView



as\_view是View基类提供的类方法，用于dispatch,根据请求方法，调用相应的方法处理请求，如发送了GET请求，那么在View()中就会分发到get()方法中处理

TemplateView源代码

**class TemplateView**(TemplateResponseMixin, ContextMixin, View)**:** *"""  
 A view that renders a template. This view will also pass into the context  
 any keyword arguments passed by the URLconf.  
 """* **def get**(self, *request*, *\*args*, *\*\*kwargs*)**:** context **=** self.get\_context\_data(**\*\****kwargs*)  
 **return** self.render\_to\_response(context)

as\_view()使用

urlpatterns = [  
 url(r'^$', views.index, name='index'),  
 url(r'^about/', TemplateView.as\_view(template\_name="about.html"))  
]

class-base-view 有两种方法

1. as\_view()传入参数，它会覆盖该view的属性，适合处理不复杂的情况
2. 继承该view,重写其方法或属性

使用方法：

urlpatterns = [

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

# url(r'^about/', TemplateView.as\_view(template\_name="about.html"))

url(r'^about/', views.AboutView.as\_view(), name='about')

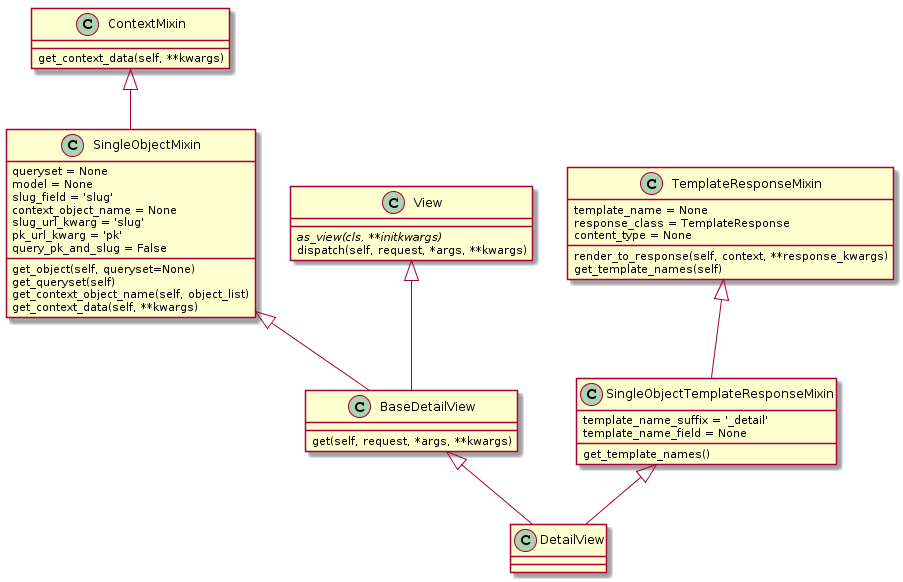
]

note:要用as\_view()调用

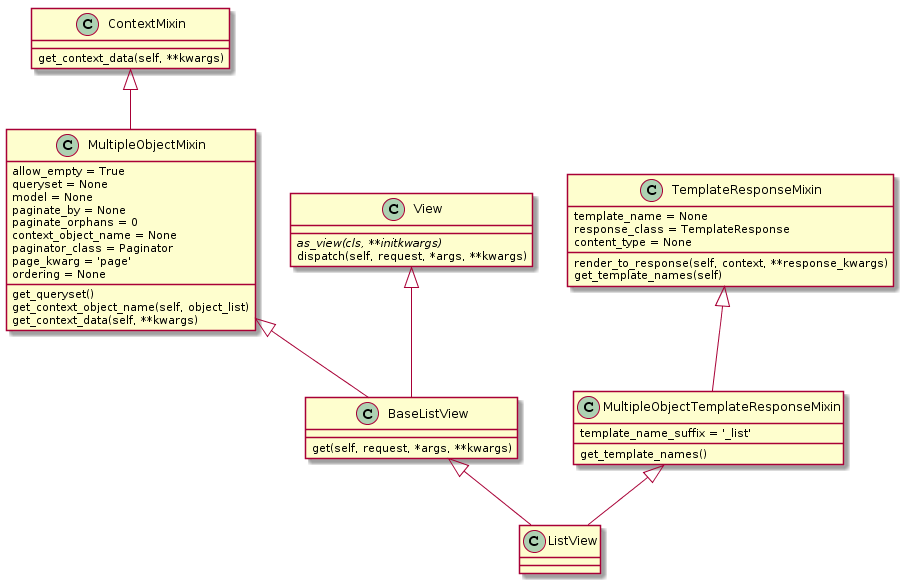
### RedirectView

### FormView

### DetailView



### ListView



## Model

### model filter 查找数据时使用正则

用iregex 可以通过正则表达式配置出类似 \*abc\*ccc\*oo这样的记录

例子：

实际项目中有这样的需求，前端输入框可以让用户通过\*号wildcard查找数据

以前django1.7版本可以安装一个django\_like的库解决，但1.8以上则不行，

经常查找相差文档，原来mysql可以通过正则来查询数据，那么django也支持这样的表达式，就是regex，只需要将用户的\*号替换一下就好了，

if ‘\*’ in find\_string:

Server.objects.filter(name\_\_iregex=name.replace(‘\*’, ‘[0-9a-z.\_]\*’)

else:

Server.objects.filter(name\_\_icontains=name)

## Form

### 初始化字段

在init() 方法里，判断instance 是否是一个实例，instance 可以是外部传递进来，如

MyForm(instance=obj),如果是实例，则

self.fields[‘myField’].initial = set\_field\_val

### 为ChoiceField设置choices

和上面的示例一样，在init里，设置self.fields[‘myField’].choices = choice\_list

# 第三方工具库

## Django-Rest-Framework

### viewset

viewset 类与view基本是相同的，但提供的动作是read或update,而不是http动作get或put.