

CCORZ

大音希声，小逼无形

博客园 首页 新随笔 联系 订阅 管理

随笔-161 文章-0 评论-16

公告

昵称：ccorz  
园龄：2年3个月  
粉丝：76  
关注：12  
+加关注

<	2018年8月						>
日	一	二	三	四	五	六	
29	30	31	1	2	3	4	
5	6	7	8	9	10	11	
12	13	14	15	16	17	18	
19	20	21	22	23	24	25	
26	27	28	29	30	31	1	
2	3	4	5	6	7	8	

搜索

找找看

谷歌搜索

常用链接

我的随笔  
我的评论  
我的参与  
最新评论  
我的标签

我的标签

python知识点(54)  
python之web框架(24)  
前端知识(16)  
Django之奇技淫巧(13)  
Tornado(4)  
python(3)  
Django RESTful framework(3)  
python之数据库操作(3)  
python之socket(2)  
python之缓存(2)  
更多

Django REST framework 的快速入门教程

我们将创建一个简单的API，让管理员用户能查看，修改系统中的用户和组。

项目搭建

创建一个新的Django项目，叫做 `tutorial`，然后开始一个新的app叫做 `quickstart`。

```
# 创建项目的目录
mkdir tutorial
cd tutorial

# 创建一个虚拟环境 (virtualenv)，来隔离地搭依赖包 (开发环境)
virtualenv env
source env/bin/activate # 在windows环境下，我们使用
`env\Scripts\activate`

# 安装Django 和 Django REST framework 到虚拟环境 (virtualenv) 中
pip install django
pip install djangorestframework

# 建立新项目和一个应用
django-admin.py startproject tutorial . # 注意末尾的'.'符号
cd tutorial
django-admin.py startapp quickstart
cd ..
```

同步数据库：

```
python manage.py migrate
```

我们也创建初始化的用户，叫做 `admin`，密码为 `password123`。稍后的案例中，我们将以该用户来登陆验证。

```
python manage.py createsuperuser
```

等你建好了数据库，创建了初始用户，一切准备完毕后，我们打开app的目录，然后开始编程啦.....

序列化器 (Serializers)

首先，我们来定义一些序列化器。我们来创建一个新的模块 (module) 叫做 `tutorial/quickstart/serializers.py`，这是我们用来描述数据是如何呈现的。

## 随笔档案

2018年5月 (2)  
 2018年4月 (3)  
 2017年7月 (1)  
 2017年6月 (7)  
 2017年4月 (1)  
 2017年3月 (9)  
 2017年2月 (28)  
 2017年1月 (4)  
 2016年12月 (3)  
 2016年11月 (9)  
 2016年10月 (9)  
 2016年9月 (13)  
 2016年8月 (15)  
 2016年7月 (14)  
 2016年6月 (25)  
 2016年5月 (18)

## 相册

able(51)  
 able1

## 常用链接

Django新手图文指导  
 jQuery语法介绍  
 免费SSL证书  
 申请免费SSL证书

## 积分与排名

积分 - 135329  
 排名 - 2404

## 最新评论

1. Re:Python之queue模块以及生产者消费者模型  
 对博主的代码我这里有两个疑问：1. 生产者消费者一直不停么；另外主线程没有对子线程进行join()，资源怎么释放呢2. 假设生产者已经不再生产了，而此时q.empty() == True的话，那么多个.....  
 --generalibm
2. Re:教你如何在linux上装逼，shell中颜色的设置  
 方法很好，可是这位大佬，您的颜色有点花哨啊

```
from django.contrib.auth.models import User, Group
from rest_framework import serializers

class
UserSerializer(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = User
        fields = ('url', 'username', 'email', 'groups')

class
GroupSerializer(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = Group
        fields = ('url', 'name')
```

需要注意的是，我们在这个案例中，使用了超链接关系（hyperlinked relations），借助的类是 `HyperlinkedModelSerializer`。你也可以使用主键（primary key）和其它一些关系型(relationships)，但超链接（hyperlinking）是非常好的RESTful设计。

## 视图 (Views)

现在我们最好是写些视图。打开 `tutorial/quickstart/views.py` 然后打起你的键盘吧～

```
from django.contrib.auth.models import User, Group
from rest_framework import viewsets
from tutorial.quickstart.serializers import UserSerializer, GroupSerializer

class UserViewSet(viewsets.ModelViewSet):
    """
    API端：允许查看和编辑用户
    """
    queryset = User.objects.all().order_by('-date_joined')
    serializer_class = UserSerializer

class GroupViewSet(viewsets.ModelViewSet):
    """
    API端：允许查看和编辑组
    """
    queryset = Group.objects.all()
    serializer_class = GroupSerializer
```

比起传统的做法，需要写很多视图，我们将所有的一般性行为（common behavior）组成一个 `ViewSet` 类。

如有需要，我们可很轻易的将其，拆分成数个单独的视图。但视图组（`viewsets`）能让视图的逻辑结构清晰，而且简洁。

## URLs

--gaby\_yan

### 3. Re:Python操作MySQL之SQLAlchemy

解释的比较透,学习了.

--MAX\_T

### 4. Re:django中的多级评论

这样在后端实现的递归, 会增加服务器的压力

--黄土地上的黑石头

### 5. Re:Django之model详解

Good

--者行孙某

## 阅读排行榜

1. python中os.popen, os.system()区别(20834)
2. Django权限系统auth模块详解(18373)
3. 教你如何在linux上装逼, shell中颜色的设置(12707)
4. Django中authenticate和login模块(10775)
5. Django自定义用户认证系统之自定义用户模型(10580)

## 评论排行榜

1. Day9作业:socket之FTP工具(3)
2. Python基础一(2)
3. 获取Django model中字段名,字段的verbose\_name,字段类型(2)
4. Day1作业2: 多层菜单查询(1)
5. 教你如何在linux上装逼, shell中颜色的设置(1)

## 推荐排行榜

1. Python操作MySQL之SQLAlchemy(2)
2. Python之Redis操作(2)
3. django中同通过getlist() 接收页面form的post数组(2)
4. 爬虫相关-scrapy框架介绍(2)
5. ansible常用模块(2)

好, 现在我们来装配API的URLs. 进入

tutorial/urls.py

```
from django.conf.urls import url, include
from rest_framework import routers
from tutorial.quickstart import views

router = routers.DefaultRouter()
router.register(r'users', views.UserViewSet)
router.register(r'groups', views.GroupViewSet)

# 使用自动化URL路由, 转配我们的API.
# 如有额外需要, 我也为可视化API添加了登陆URLs.
urlpatterns = [
    url(r'^$', include(router.urls)),
    url(r'^api-auth/', include('rest_framework.urls',
                               namespace='rest_framework'))
]
```

因为我们用了视图组 (viewsets) 而不是多个视图, 我们可以为我们的API自动的生成URL配置, 只需简单的将视图组 (viewsets) 注册到router类中即可。

同样的, 如果我们需要对API URLs进行单独控制, 我们可以使用普通基于类 (class based) 的视图, 并详细的配置每个URL。

最后, 我们为可视化API, 添加登陆/登出视图。这是可选的, 但对于需要登陆验证的API, 以及可视化的API却是非常的有用。

## 设置 (Settings)

我们也需要一些全局设置。我们想要分页 (pagination), 我们希望API只对管理用户开发。设置模块会在 tutorial/settings.py 中:

```
INSTALLED_APPS = (
    ...
    'rest_framework',
)

REST_FRAMEWORK = {
    'DEFAULT_PERMISSION_CLASSES': (
        'rest_framework.permissions.IsAdminUser',
    ),
    'PAGE_SIZE': 10
}
```

## 测试

现在来测试我们刚创建的API吧。让我们在命令行中, 把服务跑起来。

```
python ./manage.py runserver
```

使用API, 可以通过命令行, 一些工具比如

curl

```
bash: curl -H 'Accept: application/json; indent=4' -u
admin:password123 http://127.0.0.1:8000/users/
{
    "count": 2,
```

```
"next": null,
"previous": null,
"results": [
  {
    "email": "admin@example.com",
    "groups": [],
    "url": "http://127.0.0.1:8000/users/1/",
    "username": "admin"
  },
  {
    "email": "tom@example.com",
    "groups": [
    ],
    "url": "http://127.0.0.1:8000/users/2/",
    "username": "tom"
  }
]
```

或者同样的命令行工具

`httpie`

```
bash: http -a admin:password123
http://127.0.0.1:8000/users/

HTTP/1.1 200 OK
...
{
  "count": 2,
  "next": null,
  "previous": null,
  "results": [
    {
      "email": "admin@example.com",
      "groups": [],
      "url": "http://localhost:8000/users/1/",
      "username": "paul"
    },
    {
      "email": "tom@example.com",
      "groups": [
      ],
      "url": "http://127.0.0.1:8000/users/2/",
      "username": "tom"
    }
  ]
}
```

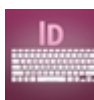
如果你借助浏览器，请确保你在右上角的入口，进行了登陆。

标签: Django RESTful framework

好文要顶

关注我

收藏该文



CCORZ

关注 - 12

粉丝 - 76

[+加关注](#)

0

0