ccorz

大音希声，小逼无形

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.....

序列器（Serializers）

首先，我们来定义一些序列器。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。

```
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 ViewSets 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

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