**Django Admin 管理工具**

Django 提供了基于 web 的管理工具。

Django自动管理工具是django.contrib的一部分。你可以在项目的 settings.py 中的INSTALLED\_APPS看到它：

INSTALLED\_APPS = (

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

)

django.contrib是一套庞大的功能集，它是Django基本代码的组成部分。

**激活管理工具**

通常我们在生成项目时会在urls.py中自动设置好，我们只需去掉注释即可。

配置项如下所示：

from django.contrib import admin

admin.autodiscover()

# And include this URLpattern...

urlpatterns = patterns('',

# ...

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

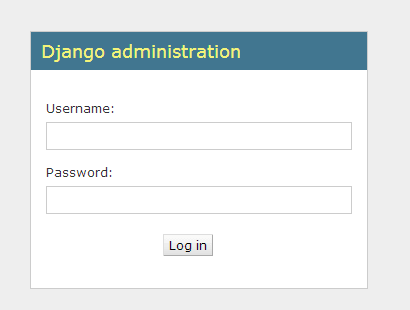
# ...

)

当这一切都配置好后，Django管理工具就可以运行了。

**使用管理工具**

启动开发服务器，然后在浏览器中访问：http://yoursite:8000/admin/，得到如下界面：



你可以通过命令 **python manage.py createsuperuser** 来创建超级用户，如下所示：

# python manage.py createsuperuser

Username (leave blank to use 'root'): admin

Email address: admin@w3cschool.cc

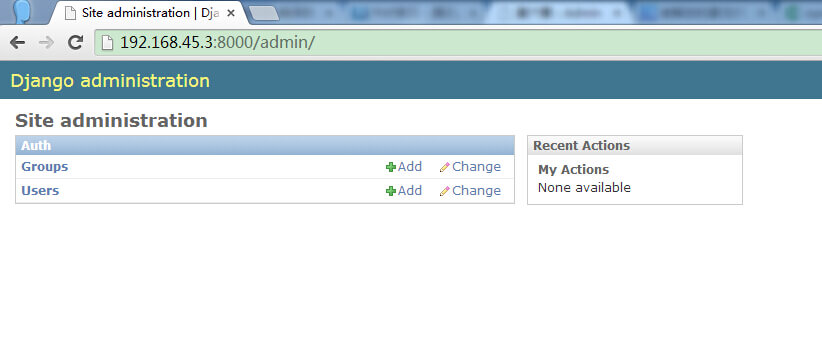
Password:

Password (again):

Superuser created successfully.

[root@solar HelloWorld]#

之后输入用户名密码登录，界面如下：



为了让admin界面管理某个数据模型，我们需要先注册该数据模型到admin。比如，我们之前在 TestModel 中已经创建了模型 Test 。修改 TestModel/admin.py:

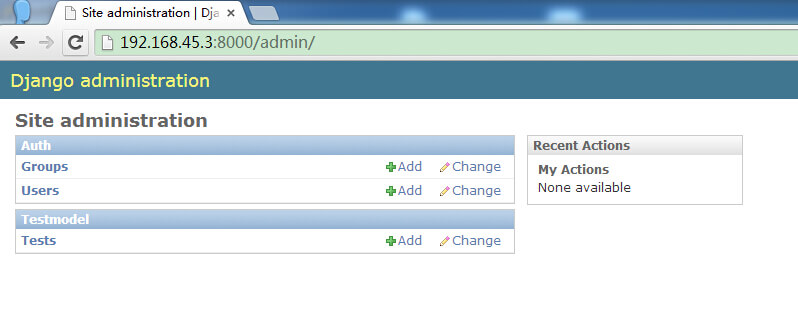
from django.contrib import admin

from TestModel.models import Test

# Register your models here.

admin.site.register(Test)

刷新后即可看到 Testmodel 数据表:



**复杂模型**

管理页面的功能强大，完全有能力处理更加复杂的数据模型。

先在 TestModel/models.py 中增加一个更复杂的数据模型：

from django.db import models

# Create your models here.

class Contact(models.Model):

name = models.CharField(max\_length=200)

age = models.IntegerField(default=0)

email = models.EmailField()

def \_\_unicode\_\_(self):

return self.name

class Tag(models.Model):

contact = models.ForeignKey(Contact)

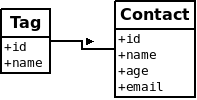
name = models.CharField(max\_length=50)

def \_\_unicode\_\_(self):

return self.name

这里有两个表。Tag以Contact为外部键。一个Contact可以对应多个Tag。

我们还可以看到许多在之前没有见过的属性类型，比如IntegerField用于存储整数。



在 TestModel/admin.py 注册多个模型并显示：

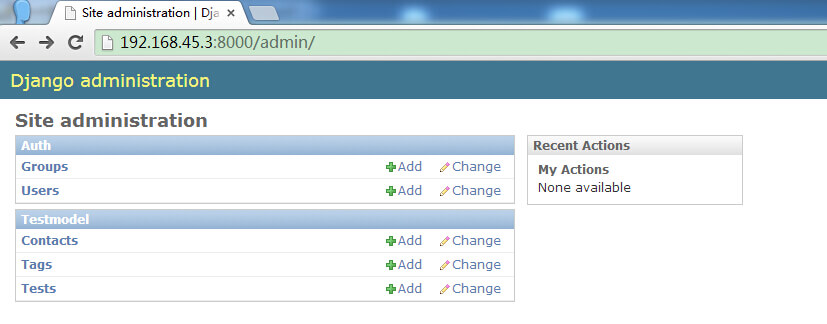
from django.contrib import admin

from TestModel.models import Test,Contact,Tag

# Register your models here.

admin.site.register([Test, Contact, Tag])

刷新管理页面，显示结果如下：



在以上管理工具我们就能进行复杂模型操作。

**自定义表单**

我们可以自定义管理页面，来取代默认的页面。比如上面的"add"页面。我们想只显示name和email部分。修改 TestModel/admin.py:

from django.contrib import admin

from TestModel.models import Test,Contact,Tag

# Register your models here.

class ContactAdmin(admin.ModelAdmin):

fields = ('name', 'email')

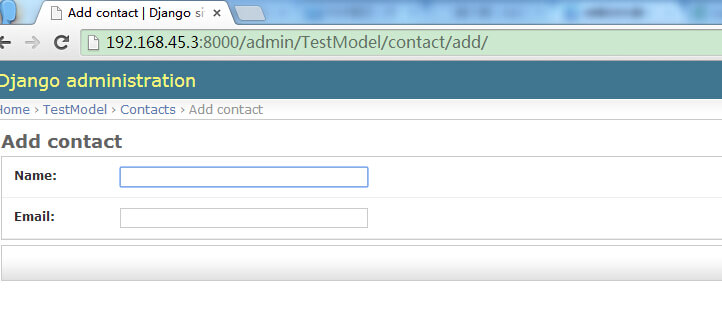
admin.site.register(Contact, ContactAdmin)

admin.site.register([Test, Tag])

以上代码定义了一个ContactAdmin类，用以说明管理页面的显示格式。

里面的fields属性定义了要显示的字段。

由于该类对应的是Contact数据模型，我们在注册的时候，需要将它们一起注册。显示效果如下：



我们还可以将输入栏分块，每个栏也可以定义自己的格式。修改TestModel/admin.py为：

from django.contrib import admin

from TestModel.models import Test,Contact,Tag

# Register your models here.

class ContactAdmin(admin.ModelAdmin):

fieldsets = (

['Main',{

'fields':('name','email'),

}],

['Advance',{

'classes': ('collapse',), # CSS

'fields': ('age',),

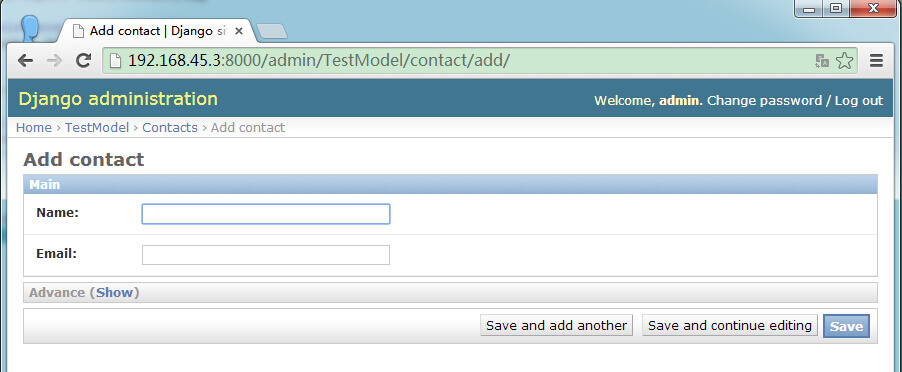
}]

)

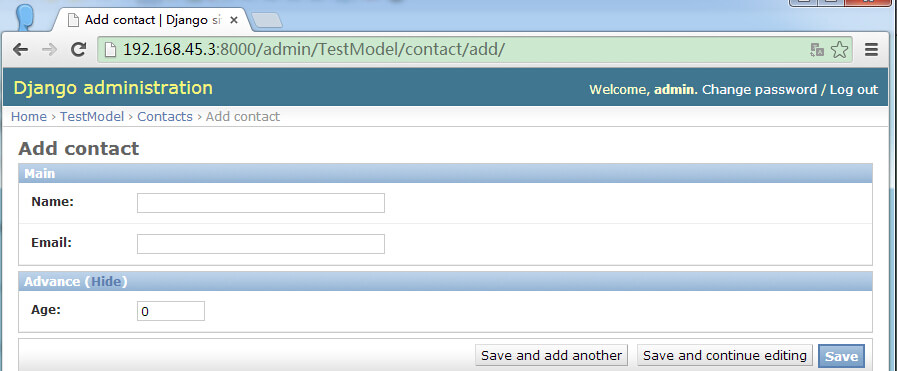
admin.site.register(Contact, ContactAdmin)

admin.site.register([Test, Tag])

上面的栏目分为了Main和Advance两部分。classes说明它所在的部分的CSS格式。这里让Advance部分隐藏：



Advance部分旁边有一个 Show 按钮，用于展开，展开后可点击 Hide 将其隐藏，如下图所示：



**内联(Inline)显示**

上面的Contact是Tag的外部键，所以有外部参考的关系。

而在默认的页面显示中，将两者分离开来，无法体现出两者的从属关系。我们可以使用内联显示，让Tag附加在Contact的编辑页面上显示。

修改TestModel/admin.py：

from django.contrib import admin

from TestModel.models import Test,Contact,Tag

# Register your models here.

class TagInline(admin.TabularInline):

model = Tag

class ContactAdmin(admin.ModelAdmin):

inlines = [TagInline] # Inline

fieldsets = (

['Main',{

'fields':('name','email'),

}],

['Advance',{

'classes': ('collapse',),

'fields': ('age',),

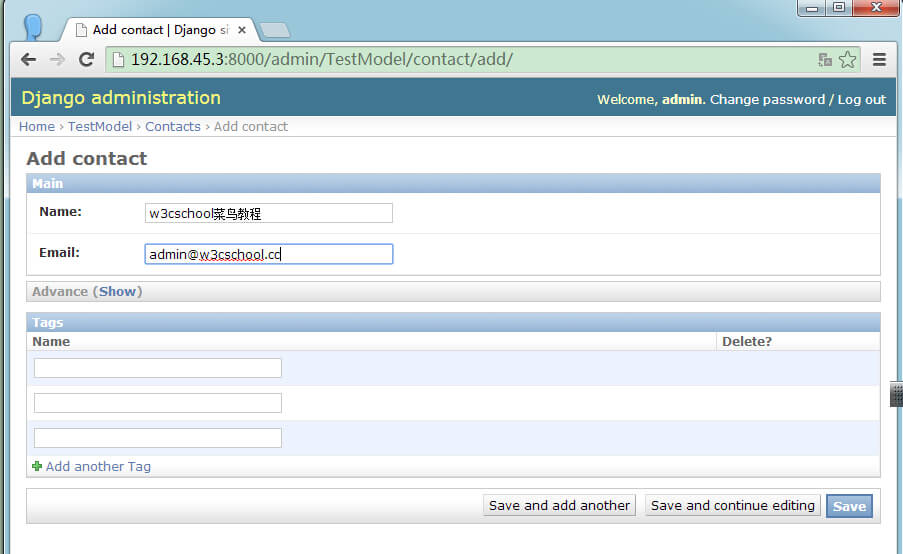
}]

)

admin.site.register(Contact, ContactAdmin)

admin.site.register([Test])

显示效果如下：



**列表页的显示**

在Contact输入数条记录后，Contact的列表页看起来如下:



我们也可以自定义该页面的显示，比如在列表中显示更多的栏目，只需要在ContactAdmin中增加list\_display属性:

from django.contrib import admin

from TestModel.models import Test,Contact,Tag

# Register your models here.

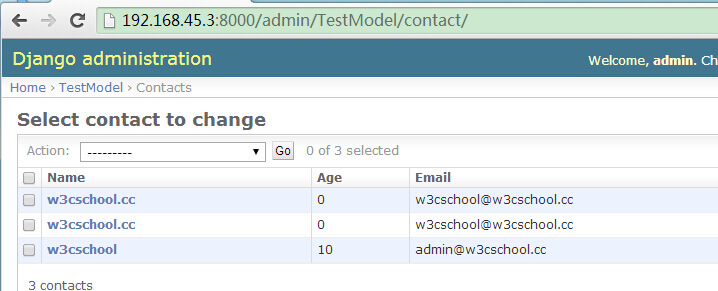
class ContactAdmin(admin.ModelAdmin):

list\_display = ('name','age', 'email') # list

admin.site.register(Contact, ContactAdmin)

admin.site.register([Test, Tag])

刷新页面显示效果如下：



搜索功能在管理大量记录时非常有，我们可以使用search\_fields为该列表页增加搜索栏：

from django.contrib import admin

from TestModel.models import Test,Contact,Tag

# Register your models here.

class ContactAdmin(admin.ModelAdmin):

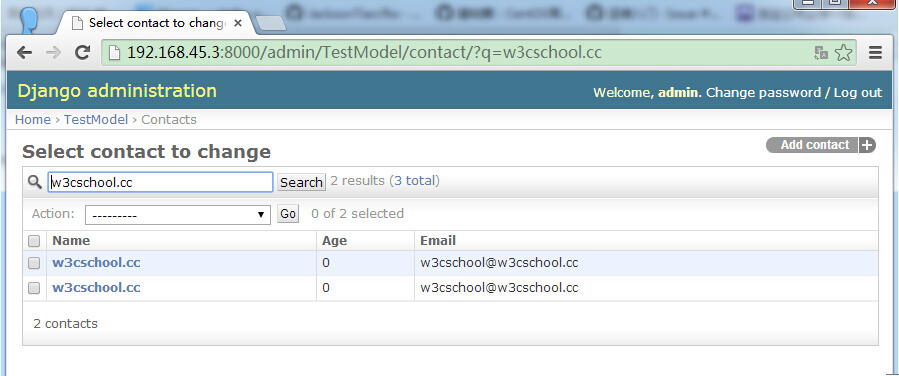
list\_display = ('name','age', 'email')

search\_fields = ('name',)

admin.site.register(Contact, ContactAdmin)

admin.site.register([Test])

在本实例中我们搜索了 name 为 w3cschool.cc(本站域名) 的记录，显示结果如下：



Django Admin 管理工具还有非常多实用的功能，感兴趣的同学可以深入研究下。