Django实例

Admin实例

- 什么是Admin

Admin是一个Django自带的一个功能强大的自动化数据管理

被授权的用户可直接在Admin中管理数据库

Django提供了许多针对Admin的定制功能

配置Admin

1、创建超级用户

python manage.py createsuperuser

2、入口地址：http://localhost:8000/admin/

3、修改settings.py中

LANGUAGE_CODE = **'zh-Hans'**

配置Admin

4、配置应用

在应用下admin.py中引入自身的models模块（或者模型类）

编辑admin.py:

admin.site.register(models.Article)

配置Admin

- 修改数据默认显示名称

  在Article类下添加一个方法

  __str__(slef)或__unicode__(self)

  如：**def** __str__(self):

  return self.title

主页面开发

- 主页面内容
  - 文章标题列表（超链接）
  - 发表博客按钮（超链接）
- 列表编写思路
  - 从数据库中取出所有的blog对象
  - 将对象打包成列表，传递到前端
  - 前端以标题超链接的形式逐个列出

主页面开发

- 模板中的for循环

  {% for xx in xxs %}

- HTML语句

  {% endfor %}

主页面开发

- views.py

```
def getbloglist(request):
    articles = models.Article.objects.all()
    return render(request, "bloglist.html", {'paramarticles': articles})
```

- **getbloglist.html**

```
{% for article in paramarticles %}
    <a href="viewblog.html">{{ article.title }}</a>
{% endfor %}
```

blog详情页

- 详情页内容
  - 标题
  - 内容
  - 修改按钮（超链接）
- 列表编写思路
  - 从数据库中取出指定的对象
  - 将对象传递到前端
  - 关键点是参数的传递

blog详情页

- views.py

```
def blog_page(request,article_id):
    article = models.Article.objects.get(pk=article_id)
    return render(request, "viewblog.html", {'paramarticle': article})
```

- blog_list.html

```
{{ paramarticle.title }}
{{ paramarticle.content }}
```

- ulrs.py 组名必须与参数名保持一致

```
url(r'^blogpage/(?P<article_id>[0-9]+)$', views.blog_page),
```

http://localhost:8000/news/blogpage/1

主页面开发（链接）

- 超链接的目标地址
  - templates用{% url ‘url_name’ param %}
  - url_name都在urls配置文件中配置
  - url(**r'^blogpage/(?P<article_id>[0-9]+)$'**, views.blog_page,name='**blog_page1**'),
  - **<a href="**{% **url** '**blog_page1**' **article.id** %}**">**

编辑页面

- 涉及到两个函数
  - 页面本身
  - submit函数