Easy 博客技术文档

目录

[命名规范 1](#_Toc506990758)

[全局介绍 1](#_Toc506990759)

[应用分类介绍 1](#_Toc506990760)

[Login\_and\_sign 1](#_Toc506990761)

[简介 1](#_Toc506990762)

[models.py 2](#_Toc506990763)

[setting.py 2](#_Toc506990764)

[urls.py 2](#_Toc506990765)

[views.py 3](#_Toc506990766)

[templates 3](#_Toc506990767)

[blogs 3](#_Toc506990768)

[简介 3](#_Toc506990769)

[static： 4](#_Toc506990770)

[templates 4](#_Toc506990771)

[func.py 4](#_Toc506990772)

[models.py 6](#_Toc506990773)

[setting.py 7](#_Toc506990774)

[urls.py 8](#_Toc506990775)

[views.py 8](#_Toc506990776)

# 命名规范

# 全局介绍

## \_\_init\_\_.py:初始化，加载数据库

## setting: 全局设置

注意事项（被修改的非默认内容，django1.11）

INSTALL\_APPS 加入 blog 和 login\_and\_sign

DATABASE: mysql数据库的内容

LANGUAGE\_CODE 由英文改为中文，便于输入中文进数据库

# 数据库设置

CREATE DATABASE blog DEFAULT CHARACTER SET utf8 COLLATE utf8\_general\_ci;

数据库创建命令，保证以utf-8为编码，可以输入中文

# 应用分类介绍

## Login\_and\_sign

### 简介

登录注册应用，用于实现注册与登录验证

### func.py

#### create\_validate\_code

##### 简介

生成可调用的验证码，参数全部设有默认，可进行相应修改，与当前应用下的template下的base内的code

#### Check\_code

##### 简介

检测验证码是否正确

### models.py

#### User

##### 字段解释

###### username: 用户名

###### password: 密码

###### email: 用户邮箱

##### Meta类

###### 用于设置后台显示的名称

##### 注

###### 字段变量verbose用于显示名称

### setting.py

#### 简介

设置各种参量

包括：密码最短长度，最大长度，用户名最大长度，邮箱最大长度，支持的邮箱类型

### urls.py

#### 简介

包括该应用下的各个页面：登录，注册，注册情况提示

### views.py

#### 简介：视图,存放返回页面主逻辑

#### login：返回登录页面

#### home：检测登入是否成功，成功跳转blogs应用的home界面，包括忘记密码，若忘记密码将会向相应邮箱投放密码验证

#### sign：返回注册界面

#### sign\_message：检验注册信息是否正确

#### create\_code\_img：验证码生成函数

#### verify：检验是否从邮箱获取信息，并更新密码

### templates

#### login\_and\_sign

该应用下各个html界面模板

## blogs

### 简介

博客内容主界面

### static：

#### 页面样式存放位置

#### 包括富文本编辑器与图片

### templates

#### base： 通用类型模板存放位置

#### blog：相应页面模板存放位置

### func.py

#### 简介：

通用类型的函数逻辑

#### passage\_test

检测文章内容是否成立，并返回带有错误信息项的字典类型context = {‘error\_message’: [ ]}, 所有错误信息以列表形式存储

并且返回所有文章内容值

#### test\_login

检验是否正常登录，获取session中的用户名，如获取失败，返回False，否则返回相应用户名 username

#### objects\_list

##### 参数要求：

request请求（与该应用中的templates的base中的page\_turning.html 配合使用， 保证不出现无法获取相应内容， request应该是以GET方法提交表单）

p\_obejects QuerySet类型的需要处理分页的内容

page\_items 每一页的内容项目数，默认为同一目录下的setting.py 的设置值

若为错误值，返回1

##### 作用：分页，计算页面值和总页数

##### 返回值：

返回一个字典类型，字典中包含以下几项

passage：该页应有的项目

index：页码

index\_all：总页面数

#### page\_turning

##### 简介：

页面翻页计算函数，通过传来的为上翻页或下翻页

返回值为当前页码根据翻页情况+或-1

### models.py

#### Passage

##### 字段解释

###### username\_id: 外键，连接User类

###### title:文章标题

###### text:文章内容

###### pub\_time:发表时间

###### type1: 标签1，用于分类类型

###### type2: 标签2

###### type3: 标签3

##### 类Meta:

###### 用于在后台的解释，名称：博客

#### Comment:

##### 字段解释

###### username\_id: 外键，连接User类

###### passage\_id: 外键，连接Passage类

###### text:文章内容

###### pub\_time:发表时间

##### 类Meta:

###### 用于在后台的解释，名称：评论

##### 注:

###### 该类的显示设置为 text的前五个字

### setting.py

#### 简介

基本常数的设置，包括：分页时每页的项数，文章标题最大值，文章内容最大值，类型1,2,3的最大字数值，评论的最大值

### urls.py

#### 该应用下的各个页面跳转

### views.py

#### home：返回首页内容（分页完成后的）

#### passage\_detail: 返回要求id的文章详细内容与评论内容

#### self\_home: 返回个人主页

#### passage\_edit: 点击修改文章后返回的内容，包括处理删除的情况

#### edit\_statu: 处理文章修改后提交的内容，若正确存储于数据库中

#### add\_passage: 返回新增文章的页面

#### add\_statu: 新增文章内容，若文章内容有所错误，返回新增文章界面（注，默认会将原本的文章填进页面内）

#### search：根据提交的情况将搜索相应的文章内容

#### add\_comment:新增评论

#### add\_comment:检测评论的正确性，并保存