达内-博客项目-Topic-API说明文档

[Python-教研部]

目录

[一， 概述：](#_Toc13914_WPSOffice_Level1) [1](#_Toc13914_WPSOffice_Level1)

[二， 事件定义：](#_Toc6753_WPSOffice_Level1) [1](#_Toc6753_WPSOffice_Level1)

[三， 开发规范:](#_Toc24725_WPSOffice_Level1) [1](#_Toc24725_WPSOffice_Level1)

[四， 数据库结构：](#_Toc11861_WPSOffice_Level1) [2](#_Toc11861_WPSOffice_Level1)

[五， 接口说明：](#_Toc3441_WPSOffice_Level1) [3](#_Toc3441_WPSOffice_Level1)

[六， 常见问题：](#_Toc23220_WPSOffice_Level1) [13](#_Toc23220_WPSOffice_Level1)

# 概述：

随着近期公司内部个人兴趣社区活动的如火如荼，为满足广大社区爱好者追求自我的梦想，特此开发内部博客；方便大家整理及自己社区活动中的一些优质内容，增加社区的互动性。此文档为发表博客相关的API详情【topic 模块】。

# 事件定义：

1. 获取博客列表 - 获取用户的博客列表，博客分为 技术和非技术两大类别；权限分为个人和公开。 其中个人博客只有在博主登录状态下才能获取
2. 发表博客 - 只有博主在登录状态下才能执行发表博客操作

# 开发规范:

1. 后端环境

Python 3.7.3 + django 1.11.8 + mysql 5.5 + Ubuntu19.04 + vim

1. 通信协议

http

1. 通信格式

json

1. API规范

一定程度上符合RESTful 定义

# 数据库结构：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 作用 | 备注1 | 备注2 |
| id | int | 主键自增 | 无 | 无 |
| title | varchar(50) | 文章主题 | 无 | 无 |
| category | varchar(20) | 博客的分类 | 1. tec 技术类型 2. no-tec 非技术类型 | 无 |
| limit | varchar(10) | 权限 | 1. public- 公开 2. private-私有 | 无 |
| introduce | varchar(90) | 博客简介 | 博客内容的前三十个字符 | 无 |
| content | text | 博客内容 | 无 | 无 |
| created\_time | datetime | 博客创建时间 | 无 | 无 |
| updated\_time | datetime | 博客修改时间 | 无 | 无 |
| author | UserProfile外键 | 无 | 无 | 无 |

# 接口说明：

## 1 发表博客

URL : http://127.0.0.1:8000/v1/topics/<username>

### 请求方式

POST

### 请求格式

json 具体参数如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| title | 用户名 | char | 必填 |
| category | 种类 | char | tec : 技术类别  No-tec : 非技术类别 |
| limit | 权限 | char | public - 公开  private - 个人 |
| content | 博客内容带HTML格式 | char | 必填 |
| content\_text | 博客纯文本格式 | char | 必填 |
|  |  |  |  |

请求示例：

{‘title: ‘haha’, ‘category’: [‘tec’,](mailto:‘abc@qq.com’,) ‘limit’: ‘public’, ‘content’: ‘abcdef<p>’ , ‘content\_text’: ‘abcdef’}

该请求需客户端在HTTP header 里添加 token, 格式如下：

Authorization ： token

### 1.3响应格式

json 具体参数如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | int | 默认正常为200，异常请见1.4 |
| username | 用户名 | char | 无 |

响应示例：

{‘code’: 200 , ‘username’: ‘abc’, }

### 1.4异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 403 | 用户未登陆 |  |

异常响应示例

{'code':403, 'error':u'Please login'}

## 2 获取用户博客列表接口

URL: http://127.0.0.1:8000/v1/topics/<username>?category=[tec|no-tec]

### 2.1 请求方式

GET

### 2.2 请求格式

2.2.1 http://127.0.0.1:8000/v1/topics/<username> 可获取用户全量数据

2.2.2 http://127.0.0.1:8000/v1/topics/<username>?category=[tec|no-tec] 可获取用户具体分类的数据， 技术(tec) 或 非技术 (no-tec)

### 2.3 响应格式

json 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | int | 请求成功，code为200，异常码请见2.4 |
| data | 返回的具体数据均在data里 | {} | 以下为data中字段 |
| nickname | 用户昵称（data中字段） | char |  |
| topics | 用户topic列表 | [] | 该集合包含{‘id’: topic\_id, ‘title’:主题，’category’: 种类【tec|no-tec】, ‘created\_time’: 创建时间, ‘introduce’: 简介, ‘author’: 作者昵称} |

响应示例：

{‘code’:200,’data’:{‘nickname’:’abc’, ’topics’:[{‘id’:1,’title’:’a’, ‘category’: ‘tec’, ‘created\_time’: ‘2018-09-03 10:30:20’, ‘introduce’: ‘aaa’, ‘author’:’abc’}]}}

### 2.4 异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 301 | 作者不存在 |  |
|  |  |  |

异常响应示例

{'code':301, 'error':u'no author'}

## 3 获取用户具体博客内容接口

URL: http://127.0.0.1:8000/v1/topics/<username>?t\_id=1111

### 3.1 请求方式

GET

### 3.2 请求格式

http://127.0.0.1:8000/v1/topics/<username> 地址后方添加查询字符串 t\_id , 值为具体博客文章的id

### 3.3 响应格式

json , 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | Int | 正常为200，异常码见3.4 |
| data | 具体数据均在data中 | {} | {‘nickname’: 作者昵称, ‘title’:博客主题, ‘category’: 博客种类[tec|no-tec], ‘created\_time’:创建时间, ‘content’: ‘具体内容’, ‘introduce’:简介, ‘next\_id’: 下一个博客的id，’next\_title’:下一个博客的主题, ‘last\_id’:上一个博客的id, ‘last\_title’: 上一个博客的主题 ‘messages’: 见下方展开, ‘message\_count’: 留言总数} |
| messages | data数据中，表示该博客的所有留言 | [] | {‘id’: 留言id,  ‘content’: 留言内容,  ‘publisher’: 留言的发布者昵称, ‘publisher\_avatar’:留言者的头像, ‘reply’: 见下方展开, ‘created\_time’: 创建时间} |
| reply | messages中，代表该留言的回复内容 | [] | {‘id’: 留言id,  ‘content’: 留言内容,  ‘publisher’: 留言的发布者昵称, ‘publisher\_avatar’:留言者的头像, ‘created\_time’:创建时间} |

响应示例：

{

"code": 200,

"data": {

"nickname": "guoxiaonao",

"title": "我的第一次",

"category": "tec",

"created\_time": "2019-06-03",

"content": "<p>我的第一次，哈哈哈哈哈<br></p>",

"introduce": "我的第一次，哈哈哈哈哈",

"author": "guoxiaonao",

"next\_id": 2,

"next\_title": "我的第二次",

"last\_id": null,

"last\_title": null,

"messages": [

{

"id": 1,

"content": "<p>写得不错啊，大哥<br></p>",

"publisher": "guoxiaonao",

"publisher\_avatar": "avatar/头像2.png",

"reply": [

{

"publisher": "guoxiaonao",

"publisher\_avatar": "avatar/头像2.png",

"created\_time": "2019-06-03 07:52:16",

"content": "谢谢您的赏识",

"msg\_id": 2

}

],

"created\_time": "2019-06-03 07:52:02"

}

],

"messages\_count": 2

}

}

### 3.4 异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 302 | 博客不存在 | 提交的查询字符串查不到topic |

异常响应示例:

{‘code’: 302, ‘error’: ‘no topic’}

## 4 删除博客接口

URL: http://127.0.0.1:8000/v1/topics/<username>?t\_id=1111

### 4.1 请求方式

DELETE

### 4.2 请求格式

http://127.0.0.1:8000/v1/topcis/<username> 地址后方添加查询字符串 t\_id , 值为具体博客文章的id

该请求需客户端在HTTP header 里添加 token, 格式如下：

Authorization ： token

### 4.3 响应格式

json, 具体参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| code | 状态 | Int | 正常为200,异常码请见4.4 |

响应示例:

{‘code’: 200}

### 4.4 异常码

|  |  |  |
| --- | --- | --- |
| 异常码 | 含义 | 备注 |
| 403 | 未登录 |  |
| 404 | URL中欲删除的用户和登陆用户不一致 |  |
| 405 | 想删除的topic不存在 |  |

异常码响应示例

{‘code’: 405, ‘error’: ‘Topic is not exist !’}

# 常见问题：

1，注意个别客户端请求需要添加 token 传回服务器端，否则异常