LeanCloud-Python-SDK Documentation

发布 *2.6.1*

asaka

2021 年 06 月 18 日

Contents

CHAPTER 1

leancloud

leancloud.**init**(*app_id*, *app_key=None*, *master_key=None*, *hook_key=None*)

初始化 LeanCloud 的 AppId / AppKey / MasterKey

参数

- **app_id** (*string_types*) – 应用的 Application ID
- **app_key** (*None or string_types*) – 应用的 Application Key
- **master_key** (*None or string_types*) – 应用的 Master Key
- **hook_key** (*None or string_type*) – application’s hook key

leancloud.**use_master_key**(*flag=True*)

是否使用 master key 发送请求。如果不调用此函数，会根据 leancloud.init 的参数来决定是否使用 master key。

leancloud.**use_production**(*flag*)

调用生产环境 / 开发环境的 cloud func / cloud hook 默认调用生产环境。

leancloud.**use_region**(*region*)

class leancloud.**FriendshipQuery**(*query_class*)

基类：leancloud.query.Query

class leancloud.**LeanCloudError**(*code*, *error*)

基类：exceptions.Exception

class leancloud.LeanCloudWarning

基类：exceptions.UserWarning

1.1 Object

class leancloud.Object(***attrs*)

基类：object

add(*attr*, *item*)

在对象此字段对应的数组末尾添加指定对象。

参数

- **attr** – 字段名
- **item** – 要添加的对象

返回 当前对象

add_unique(*attr*, *item*)

在对象此字段对应的数组末尾添加指定对象，如果此对象并没有包含在字段中。

参数

- **attr** – 字段名
- **item** – 要添加的对象

返回 当前对象

static as_class(*arg*)

bit_and(*attr*, *value*)

bit_or(*attr*, *value*)

bit_xor(*attr*, *value*)

clear()

将当前对象所有字段全部移除。

返回 当前对象

classmethod create(*class_name*, ***attributes*)

根据参数创建一个 leancloud.Object 的子类的实例化对象

参数

- **class_name** (*string_types*) – 子类名称
- **attributes** – 对象属性

返回 派生子类的实例

返回类型 *Object*

classmethod create_without_data(*id_*)

根据 objectId 创建一个 leancloud.Object，代表一个服务器上已经存在的对象。可以调用 fetch 方法来获取服务器上的数据

参数 **id** (*string_types*) – 对象的 objectId

返回 没有数据的对象

返回类型 *Object*

destroy()

从服务器上删除这个对象

返回类型 None

classmethod destroy_all(*objs*)

在一个请求中 destroy 多个 leancloud.Object 对象实例。

参数 **objs** (*list*) – 需要 destroy 的对象

disable_after_hook()

disable_before_hook()

dump()

classmethod extend(*name*)

派生一个新的 leancloud.Object 子类

参数 **name** (*string_types*) – 子类名称

返回 派生的子类

返回类型 ObjectMeta

fetch(*select=None*, *include=None*)

从服务器获取当前对象所有的值，如果与本地值不同，将会覆盖本地的值。

返回 当前对象

get(*attr*, *default=None*, *deafult=None*)

获取对象字段的值

参数 **attr** (*string_types*) – 字段名

返回 字段值

get_acl()

返回当前对象的 ACL。

返回 当前对象的 ACL

返回类型 *leancloud.ACL*

has(*attr*)

判断此字段是否有值

参数 **attr** – 字段名

返回 当有值时返回 True，否则返回 False

返回类型 bool

ignore_hook(*hook_name*)

increment(*attr*, *amount=1*)

在对象此字段上自增对应的数值，如果数值没有指定，默认为一。

参数

- **attr** – 字段名
- **amount** – 自增量

返回 当前对象

is_dirty(*attr=None*)

is_existed()

判断当前对象是否在服务器上已经存在。

返回类型 bool

is_new()

判断当前对象是否已经保存至服务器。

该方法为 SDK 内部使用（save 调用此方法 dispatch 保存操作为 REST API 的 POST 和 PUT 请求）。查询对象是否在服务器上存在请使用 is_existed 方法。

返回类型 bool

relation(*attr*)

返回对象上相应字段的 Relation

参数 **attr** (*string_types*) – 字段名

返回 Relation

返回类型 *leancloud.Relation*

remove(*attr*, *item*)

在对象此字段对应的数组中，将指定对象全部移除。

参数

- **attr** – 字段名
- **item** – 要移除的对象

返回 当前对象

save(*where=None*, *fetch_when_save=None*)
将对象数据保存至服务器

返回 None

返回类型 None

classmethod save_all(*objs*)
在一个请求中 save 多个 leancloud.Object 对象实例。

参数 **objs** (*list*) – 需要 save 的对象

set(*key_or_attrs*, *value=None*, *unset=False*)
在当前对象此字段上赋值

参数

- **key_or_attrs** (*string_types or dict*) – 字段名，或者一个包含字段名 / 值的 dict
- **value** – 字段值
- **unset** –

返回 当前对象，供链式调用

set_acl(*acl*)
为当前对象设置 ACL

返回 当前对象

unset(*attr*)
在对象上移除此字段。

参数 **attr** – 字段名

返回 当前对象

validate(*attrs*)

1.2 User

class leancloud.User(***attrs*)
基类：leancloud.object_.Object

add(*attr*, *item*)
在对象此字段对应的数组末尾添加指定对象。

参数

- **attr** – 字段名
- **item** – 要添加的对象

返回 当前对象

add_unique(*attr*, *item*)

在对象此字段对应的数组末尾添加指定对象，如果此对象并没有包含在字段中。

参数

- **attr** – 字段名
- **item** – 要添加的对象

返回 当前对象

static as_class(*arg*)

classmethod become(*session_token*)

通过 session token 获取用户对象

参数 **session_token** – 用户的 session token

返回 leancloud.User

bit_and(*attr*, *value*)

bit_or(*attr*, *value*)

bit_xor(*attr*, *value*)

classmethod change_phone_number(*sms_code*, *phone_number*)

clear()

将当前对象所有字段全部移除。

返回 当前对象

classmethod create(*class_name*, ***attributes*)

根据参数创建一个 leancloud.Object 的子类的实例化对象

参数

- **class_name** (*string_types*) – 子类名称
- **attributes** – 对象属性

返回 派生子类的实例

返回类型 *Object*

classmethod create_followee_query(*user_id*)

classmethod create_follower_query(*user_id*)

classmethod create_without_data(*id_*)

根据 objectId 创建一个 leancloud.Object，代表一个服务器上已经存在的对象。可以调用 fetch 方法来获取服务器上的数据

参数 **id** (*string_types*) – 对象的 objectId

返回 没有数据的对象

返回类型 *Object*

`destroy()`
从服务器上删除这个对象

返回类型 None

`classmethod destroy_all(objs)`
在一个请求中 destroy 多个 leancloud.Object 对象实例。

参数 **objs** (*list*) – 需要 destroy 的对象

`disable_after_hook()`

`disable_before_hook()`

`dump()`

`classmethod extend(name)`
派生一个新的 leancloud.Object 子类

参数 **name** (*string_types*) – 子类名称

返回 派生的子类

返回类型 ObjectMeta

`fetch(select=None, include=None)`
从服务器获取当前对象所有的值，如果与本地值不同，将会覆盖本地的值。

返回 当前对象

`follow(target_id)`
关注一个用户。

参数 **target_id** – 需要关注的用户的 id

`get(attr, default=None, deafult=None)`
获取对象字段的值

参数 **attr** (*string_types*) – 字段名

返回 字段值

`get_acl()`
返回当前对象的 ACL。

返回 当前对象的 ACL

返回类型 *leancloud.ACL*

`classmethod get_current()`

`get_email()`

get_mobile_phone_number()

get_roles()

get_session_token()

get_username()

has(*attr*)

判断此字段是否有值

参数 **attr** – 字段名

返回 当有值时返回 True，否则返回 False

返回类型 bool

ignore_hook(*hook_name*)

increment(*attr*, *amount=1*)

在对象此字段上自增对应的数值，如果数值没有指定，默认为一。

参数

- **attr** – 字段名
- **amount** – 自增量

返回 当前对象

is_authenticated()

判断当前用户对象是否已登录。会先检查此用户对象上是否有 *session_token*，如果有的话，会继续请求服务器验证 *session_token* 是否合法。

is_current

is_dirty(*attr=None*)

is_existed()

判断当前对象是否在服务器上已经存在。

返回类型 bool

is_linked(*provider*)

is_new()

判断当前对象是否已经保存至服务器。

该方法为 SDK 内部使用（save 调用此方法 dispatch 保存操作为 REST API 的 POST 和 PUT 请求）。查询对象是否在服务器上存在请使用 is_existed 方法。

返回类型 bool

link_with(*provider*, *third_party_auth_data*)

login(*username=None*, *password=None*, *email=None*)

登录用户。成功登录后，服务器会返回用户的 sessionToken 。

参数

- **username** – 用户名
- **email** – 邮箱地址（username 和 email 这两个参数必须传入一个且仅能传入一个）
- **password** – 用户密码

classmethod login_with(*platform*, *third_party_auth_data*)

把第三方平台号绑定到 User 上

: param platform: 第三方平台名称 base string

classmethod login_with_mobile_phone(*phone_number*, *password*)

logout()

refresh_session_token()

重置当前用户 *session token*。会使其他客户端已登录用户登录失效。

relation(*attr*)

返回对象上相应字段的 Relation

参数 **attr** (*string_types*) – 字段名

返回 Relation

返回类型 *leancloud.Relation*

remove(*attr*, *item*)

在对象此字段对应的数组中，将指定对象全部移除。

参数

- **attr** – 字段名
- **item** – 要移除的对象

返回 当前对象

classmethod request_change_phone_number(*phone_number*, *ttl=None*, *validate_token=None*)

classmethod request_email_verify(*email*)

classmethod request_login_sms_code(*phone_number*, *validate_token=None*)

classmethod request_mobile_phone_verify(*phone_number*, *validate_token=None*)

classmethod request_password_reset(*email*)

classmethod request_password_reset_by_sms_code(*phone_number*, *validate_token=None*)

classmethod reset_password_by_sms_code(*sms_code*, *new_password*)

save(*make_current=False*)

classmethod save_all(*objs*)

在一个请求中 save 多个 leancloud.Object 对象实例。

参数 **objs** (*list*) – 需要 save 的对象

session_token

set(*key_or_attrs*, *value=None*, *unset=False*)

在当前对象此字段上赋值

参数

- **key_or_attrs** (*string_types or dict*) – 字段名，或者一个包含字段名 / 值的 dict
- **value** – 字段值
- **unset** –

返回 当前对象，供链式调用

set_acl(*acl*)

为当前对象设置 ACL

返回 当前对象

classmethod set_current(*user*)

set_email(*email*)

set_mobile_phone_number(*phone_number*)

set_password(*password*)

set_username(*username*)

sign_up(*username=None*, *password=None*)

创建一个新用户。新创建的 User 对象，应该使用此方法来将数据保存至服务器，而不是使用 save 方法。用户对象上必须包含 username 和 password 两个字段

classmethod signup_or_login_with_mobile_phone(*phone_number*, *sms_code*)

param phone_nubmer: string_types param sms_code: string_types

在调用此方法前请先使用 request_sms_code 请求 sms code

unfollow(*target_id*)

取消关注一个用户。

参数 **target_id** – 需要关注的用户的 id

返回

unlink_from(*provider*)

解绑特定第三方平台

unset(*attr*)

在对象上移除此字段。

参数 **attr** – 字段名

返回 当前对象

update_password(*old_password*, *new_password*)

validate(*attrs*)

classmethod verify_mobile_phone_number(*sms_code*)

1.3 File

class leancloud.**File**(*name=u''*, *data=None*, *mime_type=None*)

基类：object

classmethod create_with_url(*name*, *url*, *meta_data=None*, *mime_type=None*)

classmethod create_without_data(*object_id*)

destroy()

fetch()

get_acl()

get_thumbnail_url(*width*, *height*, *quality=100*, *scale_to_fit=True*, *fmt=u'png'*)

metadata

mime_type

name

owner_id

query = <leancloud.query.Query object>

save()

set_acl(*acl*)

set_mime_type

size

url

1.4 Query

class leancloud.**Query**(*query_class*)

基类：object

add_ascending(*key*)

增加查询排序条件。之前指定的排序条件优先级更高。

参数 **key** – 排序字段名

返回类型 *Query*

add_descending(*key*)

增加查询排序条件。之前指定的排序条件优先级更高。

参数 **key** – 排序字段名

返回类型 *Query*

classmethod **and_**(**queries*)

根据传入的 Query 对象，构造一个新的 AND 查询。

参数 **queries** – 需要构造的子查询列表

返回类型 *Query*

ascending(*key*)

限制查询返回结果以指定字段升序排序。

参数 **key** – 排序字段名

返回类型 *Query*

contained_in(*key*, *values*)

增加查询条件，限制查询结果指定字段的值在查询值列表中

参数

- **key** – 查询条件字段名
- **values** (*list or tuple*) – 查询条件值

返回类型 *Query*

contains(*key*, *value*)

增加查询条件，限制查询结果对象指定最短的值，包含指定字符串。在数据量比较大的情况下会比较慢。

参数

- **key** – 查询条件字段名
- **value** – 需要包含的字符串

返回类型 *Query*

`contains_all`(*key*, *values*)

增加查询条件，限制查询结果指定字段的值全部包含与查询值列表中

参数

- **key** – 查询条件字段名
- **values** (*list or tuple*) – 查询条件值

返回类型 *Query*

`count`()

返回满足查询条件的对象的数量。

返回类型 int

`descending`(*key*)

限制查询返回结果以指定字段降序排序。

参数 **key** – 排序字段名

返回类型 *Query*

`classmethod do_cloud_query`(*cql*, **pvalues*)

使用 CQL 来构造查询。CQL 语法参考 这里。

参数

- **cql** – CQL 语句
- **pvalues** – 查询参数

返回类型 CQLResult

`does_not_exist`(*key*)

增加查询条件，限制查询结果对象不包含指定字段

参数 **key** – 查询条件字段名

返回类型 *Query*

`does_not_match_key_in_query`(*key*, *query_key*, *query*)

增加查询条件，限制查询结果对象指定字段的值，与另外一个查询对象的返回结果指定的值不相同。

参数

- **key** – 查询条件字段名
- **query_key** – 查询对象返回结果的字段名
- **query** (Query) – 查询对象

返回类型 *Query*

does_not_match_query(*key*, *query*)

增加查询条件，限制查询结果对象指定字段的值，与另外一个查询对象的返回结果不相同。

参数

- **key** – 查询条件字段名
- **query** (Query) – 查询对象

返回类型 *Query*

dump()

返回 当前对象的序列化结果

返回类型 dict

endswith(*key*, *value*)

增加查询条件，限制查询结果对象指定最短的值，以指定字符串结尾。在数据量比较大的情况下会比较慢。

参数

- **key** – 查询条件字段名
- **value** – 需要查询的字符串

返回类型 *Query*

equal_to(*key*, *value*)

增加查询条件，查询字段的值必须为指定值。

参数

- **key** – 查询条件的字段名
- **value** – 查询条件的值

返回类型 *Query*

exists(*key*)

增加查询条件，限制查询结果对象包含指定字段

参数 **key** – 查询条件字段名

返回类型 *Query*

find()

根据查询条件，获取包含所有满足条件的对象。

返回类型 list

first()

根据查询获取最多一个对象。

返回 查询结果

返回类型 *Object*

Raise LeanCloudError

`get`(*object_id*)

根据 objectId 查询。

参数 `object_id` – 要查询对象的 objectId

返回 查询结果

返回类型 *Object*

`greater_than`(*key*, *value*)

增加查询条件，限制查询结果指定字段的值大于查询值

参数

- `key` – 查询条件字段名
- `value` – 查询条件值

返回类型 *Query*

`greater_than_or_equal_to`(*key*, *value*)

增加查询条件，限制查询结果指定字段的值大于等于查询值

参数

- `key` – 查询条件字段名
- `value` – 查询条件值名

返回类型 *Query*

`include`(**keys*)

指定查询返回结果中包含关联表字段。

参数 `keys` – 关联子表字段名

返回类型 *Query*

`include_acl`(*value=True*)

设置查询结果的对象，是否包含 ACL 字段。需要在控制台选项中开启对应选项才能生效。

参数 `value` (*bool*) – 是否包含 ACL，默认为 True

返回类型 *Query*

`less_than`(*key*, *value*)

增加查询条件，限制查询结果指定字段的值小于查询值

参数

- `key` – 查询条件字段名
- `value` – 查询条件值

返回类型 *Query*

`less_than_or_equal_to`(*key*, *value*)

增加查询条件，限制查询结果指定字段的值小于等于查询值

参数

- **key** – 查询条件字段名
- **value** – 查询条件值

返回类型 *Query*

`limit`(*n*)

设置查询返回结果的数量。如果不设置，默认为 100。最大返回数量为 1000，如果超过这个数量，需要使用多次查询来获取结果。

参数 **n** – 限制结果的数量

返回类型 *Query*

`matched`(*key*, *regex*, *ignore_case=False*, *multi_line=False*)

增加查询条件，限制查询结果对象指定字段满足指定的正则表达式。

参数

- **key** – 查询条件字段名
- **regex** – 查询正则表达式
- **ignore_case** – 查询是否忽略大小写，默认不忽略
- **multi_line** – 查询是否匹配多行，默认不匹配

返回类型 *Query*

`matches_key_in_query`(*key*, *query_key*, *query*)

增加查询条件，限制查询结果对象指定字段的值，与另外一个查询对象的返回结果指定的值相同。

参数

- **key** – 查询条件字段名
- **query_key** – 查询对象返回结果的字段名
- **query** (Query) – 查询对象

返回类型 *Query*

`matches_query`(*key*, *query*)

增加查询条件，限制查询结果对象指定字段的值，与另外一个查询对象的返回结果相同。

参数

- **key** – 查询条件字段名
- **query** (Query) – 查询对象

返回类型 *Query*

`near`(*key*, *point*)

增加查询条件，限制返回结果指定字段值的位置与给定地理位置临近。

参数

- **key** – 查询条件字段名
- **point** – 需要查询的地理位置

返回类型 *Query*

`not_contained_in`(*key*, *values*)

增加查询条件，限制查询结果指定字段的值不在查询值列表中

参数

- **key** – 查询条件字段名
- **values** (*list or tuple*) – 查询条件值

返回类型 *Query*

`not_equal_to`(*key*, *value*)

增加查询条件，限制查询结果指定字段的值与查询值不同

参数

- **key** – 查询条件字段名
- **value** – 查询条件值

返回类型 *Query*

`classmethod or_`(**queries*)

根据传入的 Query 对象，构造一个新的 OR 查询。

参数 queries – 需要构造的子查询列表

返回类型 *Query*

`scan`(*batch_size=None*, *scan_key=None*)

`select`(**keys*)

指定查询返回结果中只包含某些字段。可以重复调用，每次调用的包含内容都将会被返回。

参数 keys – 包含字段名

返回类型 *Query*

`size_equal_to`(*key*, *size*)

增加查询条件，限制查询结果指定数组字段长度与查询值相同

参数

- **key** – 查询条件数组字段名

- **size** – 查询条件值

返回类型 *Query*

skip(*n*)

查询条件中跳过指定个数的对象，在做分页时很有帮助。

参数 **n** – 需要跳过对象的个数

返回类型 *Query*

startswith(*key*, *value*)

增加查询条件，限制查询结果对象指定最短的值，以指定字符串开头。在数据量比较大的情况下会比较慢。

参数

- **key** – 查询条件字段名
- **value** – 需要查询的字符串

返回类型 *Query*

within_geo_box(*key*, *southwest*, *northeast*)

增加查询条件，限制返回结果指定字段值的位置在指定坐标范围之内。

参数

- **key** – 查询条件字段名
- **southwest** – 限制范围西南角坐标
- **northeast** – 限制范围东北角坐标

返回类型 *Query*

within_kilometers(*key*, *point*, *max_distance*, *min_distance=None*)

增加查询条件，限制返回结果指定字段值的位置在某点的一段距离之内。

参数

- **key** – 查询条件字段名
- **point** – 查询地理位置
- **max_distance** – 最大距离限定（千米）
- **min_distance** – 最小距离限定（千米）

返回类型 *Query*

within_miles(*key*, *point*, *max_distance*, *min_distance=None*)

增加查询条件，限制返回结果指定字段值的位置在某点的一段距离之内。

参数

- **key** – 查询条件字段名

- **point** – 查询地理位置
- **max_distance** – 最大距离限定（英里）
- **min_distance** – 最小距离限定（英里）

返回类型 *Query*

within_radians(*key*, *point*, *max_distance*, *min_distance=None*)

增加查询条件，限制返回结果指定字段值的位置在某点的一段距离之内。

参数

- **key** – 查询条件字段名
- **point** – 查询地理位置
- **max_distance** – 最大距离限定（弧度）
- **min_distance** – 最小距离限定（弧度）

返回类型 *Query*

1.5 Relation

class leancloud.**Relation**(*parent*, *key=None*)

基类：object

add(**obj_or_objs*)

添加一个新的 leancloud.Object 至 Relation。

参数 **obj_or_objs** – 需要添加的对象或对象列表

dump()

query

获取指向 Relation 内容的 Query 对象。

返回类型 *leancloud.Query*

remove(**obj_or_objs*)

从一个 Relation 中删除一个 leancloud.Object 。

参数 **obj_or_objs** – 需要删除的对象或对象列表

返回

classmethod reverse_query(*parent_class*, *relation_key*, *child*)

创建一个新的 Query 对象，反向查询所有指向此 Relation 的父对象。

参数

- **parent_class** – 父类名称

- **relation_key** – 父类中 Relation 的字段名
- **child** – 子类对象

返回 leancloud.Query

1.6 Role

class leancloud.Role(*name=None*, *acl=None*)

基类：leancloud.object_.Object

add(*attr*, *item*)

在对象此字段对应的数组末尾添加指定对象。

参数

- **attr** – 字段名
- **item** – 要添加的对象

返回 当前对象

add_unique(*attr*, *item*)

在对象此字段对应的数组末尾添加指定对象，如果此对象并没有包含在字段中。

参数

- **attr** – 字段名
- **item** – 要添加的对象

返回 当前对象

static as_class(*arg*)

bit_and(*attr*, *value*)

bit_or(*attr*, *value*)

bit_xor(*attr*, *value*)

clear()

将当前对象所有字段全部移除。

返回 当前对象

classmethod create(*class_name*, ***attributes*)

根据参数创建一个 leancloud.Object 的子类的实例化对象

参数

- **class_name** (*string_types*) – 子类名称
- **attributes** – 对象属性

返回 派生子类的实例

返回类型 *Object*

classmethod create_without_data(*id_*)

根据 objectId 创建一个 leancloud.Object，代表一个服务器上已经存在的对象。可以调用 fetch 方法来获取服务器上的数据

参数 **id** (*string_types*) – 对象的 objectId

返回 没有数据的对象

返回类型 *Object*

destroy()

从服务器上删除这个对象

返回类型 None

classmethod destroy_all(*objs*)

在一个请求中 destroy 多个 leancloud.Object 对象实例。

参数 **objs** (*list*) – 需要 destroy 的对象

disable_after_hook()

disable_before_hook()

dump()

classmethod extend(*name*)

派生一个新的 leancloud.Object 子类

参数 **name** (*string_types*) – 子类名称

返回 派生的子类

返回类型 ObjectMeta

fetch(*select=None*, *include=None*)

从服务器获取当前对象所有的值，如果与本地值不同，将会覆盖本地的值。

返回 当前对象

get(*attr*, *default=None*, *deafult=None*)

获取对象字段的值

参数 **attr** (*string_types*) – 字段名

返回 字段值

get_acl()

返回当前对象的 ACL。

返回 当前对象的 ACL

返回类型 *leancloud.ACL*

`get_name()`
获取 Role 的 name，等同于 role.get('name')

`get_roles()`

`get_users()`
获取当前 Role 下所有绑定的用户。

`has(attr)`
判断此字段是否有值

参数 **attr** – 字段名

返回 当有值时返回 True，否则返回 False

返回类型 bool

`ignore_hook(hook_name)`

`increment(attr, amount=1)`
在对象此字段上自增对应的数值，如果数值没有指定，默认为一。

参数

- **attr** – 字段名
- **amount** – 自增量

返回 当前对象

`is_dirty(attr=None)`

`is_existed()`
判断当前对象是否在服务器上已经存在。

返回类型 bool

`is_new()`
判断当前对象是否已经保存至服务器。

该方法为 SDK 内部使用（save 调用此方法 dispatch 保存操作为 REST API 的 POST 和 PUT 请求）。查询对象是否在服务器上存在请使用 is_existed 方法。

返回类型 bool

`name`

`relation(attr)`
返回对象上相应字段的 Relation

参数 **attr** (*string_types*) – 字段名

返回 Relation

返回类型 *leancloud.Relation*

remove(*attr*, *item*)

在对象此字段对应的数组中，将指定对象全部移除。

参数

- **attr** – 字段名
- **item** – 要移除的对象

返回 当前对象

roles

save(*where=None*, *fetch_when_save=None*)

将对象数据保存至服务器

返回 None

返回类型 None

classmethod save_all(*objs*)

在一个请求中 save 多个 leancloud.Object 对象实例。

参数 **objs** (*list*) – 需要 save 的对象

set(*key_or_attrs*, *value=None*, *unset=False*)

在当前对象此字段上赋值

参数

- **key_or_attrs** (*string_types or dict*) – 字段名，或者一个包含字段名 / 值的 dict
- **value** – 字段值
- **unset** –

返回 当前对象，供链式调用

set_acl(*acl*)

为当前对象设置 ACL

返回 当前对象

set_name(*name*)

为 Role 设置 name，等同于 role.set（‘name’, name)

unset(*attr*)

在对象上移除此字段。

参数 **attr** – 字段名

返回 当前对象

users

validate(*attrs*)

1.7 ACL

class leancloud.**ACL**(*permissions_by_id=None*)

基类：object

dump()

get_public_read_access()

get_public_write_access()

get_read_access(*user_id*)

get_role_read_access(*role*)

get_role_write_access(*role*)

get_write_access(*user_id*)

set_public_read_access(*allowed*)

set_public_write_access(*allowed*)

set_read_access(*user_id*, *allowed*)

set_role_read_access(*role*, *allowed*)

set_role_write_access(*role*, *allowed*)

set_write_access(*user_id*, *allowed*)

1.8 GeoPoint

class leancloud.**GeoPoint**(*latitude=0*, *longitude=0*)

基类：object

dump()

kilometers_to(*other*)

Returns the distance from this GeoPoint to another in kilometers.

参数 **other** (GeoPoint) – point the other GeoPoint

返回类型 float

latitude

当前对象的纬度

longitude

当前对象的经度

miles_to(*other*)

Returns the distance from this GeoPoint to another in miles.

参数 **other** (GeoPoint) – point the other GeoPoint

返回类型 float

radians_to(*other*)

Returns the distance from this GeoPoint to another in radians.

参数 **other** (GeoPoint) – point the other GeoPoint

返回类型 float

1.9 Engine

class leancloud.**Engine**(*wsgi_app=None*, *fetch_user=True*)

基类: object

LeanEngine middleware.

after_delete(**args*, ***kwargs*)

after_save(**args*, ***kwargs*)

after_update(**args*, ***kwargs*)

before_delete(**args*, ***kwargs*)

before_save(**args*, ***kwargs*)

before_update(**args*, ***kwargs*)

define(**args*, ***kwargs*)

on_bigquery(**args*, ***kwargs*)

on_insight(**args*, ***kwargs*)

on_login(**args*, ***kwargs*)

on_verified(**args*, ***kwargs*)

register(*engine*)

run(**args*, ***kwargs*)

start()

stop()

wrap(*wsgi_app*)

1.10 HttpsRedirectMiddleware

class leancloud.engine.HttpsRedirectMiddleware(*wsgi_app*)

基类：object

1.11 CookieSessionMiddleware

class leancloud.engine.CookieSessionMiddleware(*app*, *secret*, *name=u'leancloud:session'*, *excluded_paths=None*, *fetch_user=False*, *expires=None*, *max_age=None*)

基类：object

用来在 webhosting 功能中实现自动管理 LeanCloud 用户登录状态的 WSGI 中间件。使用此中间件之后，在处理 web 请求中调用了 *leancloud.User.login()* 方法登录成功后，会将此用户 session token 写入到 cookie 中。后续此次会话都可以通过 *leancloud.User.get_current()* 获取到此用户对象。

参数

- **secret** (*str*) – 对保存在 cookie 中的用户 session token 进行签名时需要的 key，可使用任意方法随机生成，请不要泄漏
- **name** (*str*) – 在 cookie 中保存的 session token 的 key 的名称，默认为 “leancloud:session”
- **excluded_paths** (*list*) – 指定哪些 URL path 不处理 session token，比如在处理静态文件的 URL path 上不进行处理，防止无谓的性能浪费
- **fetch_user** (*bool*) – 处理请求时是否要从存储服务获取用户数据，如果为 false 的话，leancloud.User.get_current() 获取到的用户数据上除了 session_token 之外没有任何其他数据，需要自己调用 fetch() 来获取。为 true 的话，会自动在用户对象上调用 fetch()，这样将会产生一次数据存储的 API 调用。默认为 false
- **expires** (*int or datetime*) – 设置 cookie 的 expires
- **max_age** (*int*) – 设置 cookie 的 max_age，单位为秒

post_process(*environ*, *headers*)

pre_process(*environ*)

CHAPTER 2

leancloud.push

class leancloud.push.**Installation**(***attrs*)

基类：leancloud.object_.Object

class leancloud.push.**Notification**(***attrs*)

基类：leancloud.object_.Object

fetch(**args*, ***kwargs*)

同步服务器的 Notification 数据

save(**args*, ***kwargs*)

leancloud.push.**send**(*data*, *channels=None*, *push_time=None*, *expiration_time=None*, *expiration_interval=None*, *where=None*, *cql=None*, *flow_control=None*, *prod=None*)

发送推送消息。返回结果为此条推送对应的 _Notification 表中的对象，但是如果需要使用其中的数据，需要调用 fetch() 方法将数据同步至本地。

参数

- **channels** (*list or tuple*) – 需要推送的频道
- **push_time** (*datetime*) – 推送的时间
- **expiration_time** (*datetime*) – 消息过期的绝对日期时间
- **expiration_interval** (*int*) – 消息过期的相对时间，从调用 API 的时间开始算起，单位是秒

- **where** (leancloud.Query) – 一个查询 _Installation 表的查询条件 leancloud.Query 对象
- **cql** (*string_types*) – 一个查询 _Installation 表的查询条件 CQL 语句
- **data** – 推送给设备的具体信息，详情查看 https://leancloud.cn/docs/push_guide.html# 消息内容 _Data
- **flow_control** – 不为 None 时开启平滑推送，值为每秒推送的目标终端用户数。开启时指定低于 1000 的值，按 1000 计。
- **prod** – 仅对 iOS 推送有效，设置将推送发至 APNs 的开发环境（dev）还是生产环境（prod）。

返回类型 *Notification*

Type flow_control: int

Type prod: string

CHAPTER 3

leancloud.cloud

class leancloud.cloud.**Captcha**(*token*, *url*)

基类：object

表示图形验证码

verify(*code*)

验证用户输入与图形验证码是否匹配:params code: 用户填写的验证码

leancloud.cloud.**get_server_time**()

leancloud.cloud.**request_captcha**(*size=None*, *width=None*, *height=None*, *ttl=None*)

请求生成新的图形验证码:return: Captcha

leancloud.cloud.**request_sms_code**(*phone_number*, *idd=u'+86'*, *sms_type=u'sms'*, *validate_token=None*, *template=None*, *sign=None*, *params=None*)

请求发送手机验证码:param phone_number: 需要验证的手机号码:param idd: 号码的所在地国家代码，默认为中国（+86）:param sms_type: 验证码发送方式，' voice' 为语音，' sms' 为短信:param template: 模版名称:param sign: 短信签名名称:return: None

leancloud.cloud.**rpc**(*_cloud_rpc_name*, ***params*)

调用 LeanEngine 上的远程代码与 cloud.run 类似，但是允许传入 leancloud.Object 作为参数，也允许传入 leancloud.Object 作为结果:param name: 需要调用的远程 Cloud Code 的名称:type name: basestring :param params: 调用参数:return: 调用结果

leancloud.cloud.**run**(*_cloud_func_name*, ***params*)

调用 LeanEngine 上的远程代码:param name: 需要调用的远程 Cloud Code 的名称:type name: string_types :param params: 调用参数:return: 调用结果

leancloud.cloud.**verify_captcha**(*code*, *token*)

验证用户输入与图形验证码是否匹配:params code: 用户填写的验证码:params token: 图形验证码对应的 token :return: validate token

leancloud.cloud.**verify_sms_code**(*phone_number*, *code*)

获取到手机验证码之后，验证验证码是否正确。如果验证失败，抛出异常。:param phone_number: 需要验证的手机号码:param code: 接受到的验证码:return: None

CHAPTER 4

Indices and tables

- genindex
- search

Python 模块索引

l

索引

Q

R

S

U

V

W