Catalist Documentation

Release 0.0

Rachel Wu, Tony Zhang

January 26, 2016

Contents

1 catalist package **3**
- 1.1 Submodules . . . 3
- 1.2 catalist.api module . . . 3
  - 1.2.1 catalist.api . . . 3
- 1.3 catalist.database module . . . 6
  - 1.3.1 catalist.database . . . 6
- 1.4 catalist.errorviews module . . . 10
- 1.5 catalist.permissions module . . . 11
  - 1.5.1 catalist.permissions . . . 11
- 1.6 catalist.views module . . . 11
  - 1.6.1 catalist.views . . . 11
- 1.7 Module contents . . . 12

2 Indices and tables **13**

Python Module Index **15**

Index **17**

Contents:

CHAPTER 1

catalist package

1.1 Submodules

1.2 catalist.api module

1.2.1 catalist.api

This module implements our web app's API. All operations we'll probably need are here (however poorly written they may be). Relevant permissions are required everywhere.

copyright

3. 2016 Rachel Wu, Tony Zhang

license lol don't steal this code

exception `catalist.api.`**`InvalidAPIUsage`**(*message*, *status_code=None*, *payload=None*)
Bases: `exceptions.Exception`

A class for exceptions to raise in invalid API usage. Shamelessly pillaged from Flask's documentation

`status_code` = 400

`to_dict`()

`catalist.api.`**`add_to_my_lists`**()
Add a specified list to "My Lists".

POST: listid: <listid>

`catalist.api.`**`autocomplete`**()
completes a word fragment with a possible list type usage: POST to this route with {"fragment": myfragment}, response is the list of possible completions of *myfragment* drawn from *autocomplete_dict*

`catalist.api.`**`autocomplete_user`**()

`catalist.api.`**`create_list`**()
Initialize a new, empty list and return the assigned listid.

`catalist.api.`**`entry_delete`**()
Delete an entry from a Catalist. Requires at least edit permission.

POST: listid: the id of the Catalist entryind: the index of the entry to remove

catalist.api.**entry_title_save**()
AJAXily save the title of an entry. Requires at least edit permission

usage: POST a JS associative array (basically a dict) like so: {

listid: <the list id>, entryind: <index of entry w.r.t. list (0-indexing)>, newvalue: <new entry title>

}

catalist.api.**get_list_perms**()
Returns a list of editors and viewers for the current list.

GET: listid: the relevant listid

catalist.api.**get_pref**(**args*, ***kwargs*)
Get the preferred theme for the current user.

Returns the preferred theme [0, 1, ..]

Return type theme

catalist.api.**handle_invalid_usage**(*error*)

catalist.api.**key_save**()
Save a key. Requires at least edit permission.

POST a JS associative array (basically a dict) like so: {

listid: <the list id>, entryind: <index of entry w.r.t. list (0-indexing)>, index: <index of key-val pair w.r.t. entry>, newvalue: <new value of key>

}

catalist.api.**key_val_save**(*req_form*, *key_or_val*)
Save a key or value in a KVP. Auxillary method for */api/savekey* and */api/savevalue* – captures repetitive code.

catalist.api.**kvp_delete**()
Delete a key-value pair from a Catalist entry. Requires at least edit permission.

POST: listid: the id of the Catalist entryind: the index of the entry to remove index: the index of the kvp within the entry

catalist.api.**list_delete**()
Delete a Catalist. Requires at least own permission

usage: POST a JSON associative array as follows: {

listid: <the id of the list to be deleted>

}

catalist.api.**list_save**()
Save an entire list. If listid is provided, the list is written onto the referenced list. Otherwise, a new list is created. In both cases the listid to which we saved the list is returned.

POST: title: the-title, contents: [[entry-title-0, [[key, value], ...]] ...] listid: the-listid-to-save-to (optional)

Returns The given or assigned listid

catalist.api.**list_title_save**()
AJAXily save the title of a Catalist ^_^ Requires at least edit permission.

usage: POST a JS assoc array like so: {

listid: <the list id>, newvalue: <our new title>

}

catalist.api.**logged_in**()
Return loggedin=1 if logged in, else 0.

catalist.api.**make_list**()
Create a new, blank list, initializing it with appropriate values and assigning it a creator/owner, if the current user is authenticated.

Returns A response containing a dict *{'listid': assigned-list-id}*

catalist.api.**my_lists_interact**(*listid*, *addQ*)
Add or remove a list with specified listid from "My Lists".

Parameters

- **listid** – the listid of the list
- **addQ** – an integer specifying whether to add or remove: 1 to add, -1 to remove

catalist.api.**permissions_forfeit**()
Forfeit permissions to a list. Effectively sets the user's permission to *Catalist.public_level*.

POST: listid: the relevant listid

catalist.api.**permissions_get**()
Get the permission level the current user has for a particular list.

POST: listid: the relevant listid

Returns the user's current permission level for the list.

Return type permission

catalist.api.**permissions_set**()
Set permissions for a user. Requires at least own permission on the relevant Catalist.

POST: listid: the relevant listid [target]: the username of the user whose permissions we'd

like to change. Defaults to current user if not specified.

permission: one of {none | view | edit | own | admin}

catalist.api.**public_level_get**()
Get the permission level for a list for the public at-large.

POST: listid: the listid

catalist.api.**public_level_set**()
Set the permission level for a list for the public at-large. Requires own permission.

POST: listid: the relevant listid permission: one of {none | view | edit}

catalist.api.**remove_from_my_lists**()
Remove a specified list from "My Lists".

POST: listid: <listid>

catalist.api.**value_save**()
Save the value in a particular key-value pair. Requires at least edit permission.

The API is virtually identical the that of key_save()

catalist.api.**vote**()
Query or submit vote information for a particular Catalist entry. In the former mode of operation, return the

user's current vote and the entry's current score. In the latter, return the user's new vote and the entry's new score.

Requires at least view permission to the relevant Catalist.

POST: listid: the listid of the relevant Catalist entryind: the index of the entry we're voting on vote: either 0, 1, -1, or 100, depending on what we're doing–

100 to simply query vote information, -1, 0, or 1 for downvote, no vote, or upvote, respectively

Returns the user's current (or new) vote score: the entry's current (or new) score

Return type current_vote

1.3 catalist.database module

1.3.1 catalist.database

This module defines our database schemata.

copyright

3. 2016 Rachel Wu, Tony Zhang

license lol don't steal this code pls

class `catalist.database.`**`Catalist`**(**args*, ***values*)
Bases: `flask_mongoengine.Document`

A class for our lists (Catalists :P)

exception `DoesNotExist`
Bases: `mongoengine.errors.DoesNotExist`

exception `Catalist.`**`MultipleObjectsReturned`**
Bases: `mongoengine.errors.MultipleObjectsReturned`

`Catalist.`**`contents`**
A `ListField` designed specially to hold a list of embedded documents to provide additional query helpers.

Note: The only valid list values are subclasses of `EmbeddedDocument`.

New in version 0.9.

`Catalist.`**`created`**
A datetime field.

Uses the python-dateutil library if available alternatively use time.strptime to parse the dates. Note: python-dateutil's parser is fully featured and when installed you can utilise it to convert varying types of date formats into valid python datetime objects.

Note: Microseconds are rounded to the nearest millisecond. Pre UTC microsecond support is effectively broken. Use `ComplexDateTimeField` if you need accurate microsecond support.

`Catalist.`**`creator`**
A unicode string field.

Catalist.**editors**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

Catalist.**id**
A field wrapper around MongoDB's ObjectIds.

Catalist.**last_visited**
A datetime field.

Uses the python-dateutil library if available alternatively use time.strptime to parse the dates. Note: python-dateutil's parser is fully featured and when installed you can utilise it to convert varying types of date formats into valid python datetime objects.

Note: Microseconds are rounded to the nearest millisecond. Pre UTC microsecond support is effectively broken. Use `ComplexDateTimeField` if you need accurate microsecond support.

Catalist.**listid**
A unicode string field.

Catalist.**mylisters**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

Catalist.**objects = []**

Catalist.**owners**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

Catalist.**public_level**
A unicode string field.

Catalist.**title**
A unicode string field.

Catalist.**viewers**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

class catalist.database.**CatalistEntry**(**args*, ***kwargs*)
Bases: mongoengine.document.EmbeddedDocument

A class for the entries in our Catalists

contents
A ListField designed specially to hold a list of embedded documents to provide additional query helpers.

Note: The only valid list values are subclasses of EmbeddedDocument.

New in version 0.9.

downvoters
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

score
An 32-bit integer field.

title
A unicode string field.

upvoters
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

class catalist.database.**CatalistKVP**(**args*, ***kwargs*)
Bases: mongoengine.document.EmbeddedDocument

A class for individual key-value pairs in our Catalist entries

key
A unicode string field.

kvpid
A unicode string field.

value
A unicode string field.

class catalist.database.**Role**(**args*, ***values*)
Bases: flask_mongoengine.Document, flask_security.core.RoleMixin

A class for user roles (e.g. user, admin, ...)

exception DoesNotExist
Bases: mongoengine.errors.DoesNotExist

exception Role.**MultipleObjectsReturned**
Bases: mongoengine.errors.MultipleObjectsReturned

Role.**description**
A unicode string field.

Role.**id**
A field wrapper around MongoDB's ObjectIds.

Role.**name**
A unicode string field.

Role.**objects = []**

class catalist.database.**User**(**args*, ***values*)
Bases: flask_mongoengine.Document, flask_security.core.UserMixin

A class for users. Can have any/none of these attributes.

exception DoesNotExist
Bases: mongoengine.errors.DoesNotExist

exception User.**MultipleObjectsReturned**
Bases: mongoengine.errors.MultipleObjectsReturned

User.**acquaintances**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

User.**active**
A boolean field type.

New in version 0.1.2.

User.**anti_my_lists**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

Note: Required means it cannot be empty - as the default for ListFields is []

User.**confirmed_at**
A datetime field.

Uses the python-dateutil library if available alternatively use time.strptime to parse the dates. Note: python-dateutil's parser is fully featured and when installed you can utilise it to convert varying types of date formats into valid python datetime objects.

Note: Microseconds are rounded to the nearest millisecond. Pre UTC microsecond support is effectively broken. Use ComplexDateTimeField if you need accurate microsecond support.

User.**email**
A unicode string field.

User.**firstname**
A unicode string field.

User.**id**
A field wrapper around MongoDB's ObjectIds.

User.**last_active**
A datetime field.

Uses the python-dateutil library if available alternatively use time.strptime to parse the dates. Note: python-dateutil's parser is fully featured and when installed you can utilise it to convert varying types of date formats into valid python datetime objects.

Note: Microseconds are rounded to the nearest millisecond. Pre UTC microsecond support is effectively broken. Use `ComplexDateTimeField` if you need accurate microsecond support.

User.**lastname**
A unicode string field.

User.**my_custom_lists**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

---

Note: Required means it cannot be empty - as the default for ListFields is []

User.**objects** = []

User.**password**
A unicode string field.

User.**preferred_theme**
An 32-bit integer field.

User.**roles**
A list field that wraps a standard field, allowing multiple instances of the field to be used as a list in the database.

If using with ReferenceFields see: one-to-many-with-listfields

---

Note: Required means it cannot be empty - as the default for ListFields is []

User.**uid**
A unicode string field.

1.4 catalist.errorviews module

copyright

3. 2016 Rachel Wu, Tony Zhang

license lol don't steal this code pls

catalist.errorviews.**forbidden**(*e*)

catalist.errorviews.**internal_server_error**(*e*)

catalist.errorviews.**method_not_allowed**(*e*)

catalist.errorviews.**page_gone**(*e*)

catalist.errorviews.**page_not_found**(*e*)

1.5 catalist.permissions module

1.5.1 catalist.permissions

This module implements handy utilities for working with user permissions.

copyright

3. 2016 Rachel Wu, Tony Zhang

license lol don't steal this code pls

catalist.permissions.**admin_unames = ['zutara']**
Currently admins are determined by residency on this list. Hacky, I know. .__.

catalist.permissions.**cmp_permission**(*perm1*, *perm2*)
Return a positive/0/negative integer when perm1 >/=/< perm2.

catalist.permissions.**perm_list = ['none', 'view', 'edit', 'own', 'admin']**
A list of all permission levels, from lowest to highest. The levels:

1.none – no permission

2.view – permission to view a list

3.edit – permission to edit a list

4.own – permission to change permission for a list

5.admin – can do anything

catalist.permissions.**query_cur_perm**(*catalist*)
Find the permission the current user has for list *catalist*

catalist.permissions.**query_permission**(*user*, *catalist*)
Give the permission level *user* has for a list *catalist*. Handles anonymous users.

1.6 catalist.views module

1.6.1 catalist.views

This module contains most of the views (read: non-error views) for our web app.

copyright

3. 2016 Rachel Wu, Tony Zhang

license lol don't steal this code pls

catalist.views.**about**()
About us.

catalist.views.**get_id**()
Return name of current user

catalist.views.**getlist**(*listid*)
Display a list with given listid from our database.

catalist.views.**human_readable_time_since**(*tiem*)
Give a human-readable representation of time elapsed since a given time

Parameters **tiem** – a datetime object representing the given time.

catalist.views.**index**()
Our homepage!

catalist.views.**login**()
Page for user login

catalist.views.**logout**()
Log out the current user, clearing the Remember Me cookie

catalist.views.**preview_list**(*listid*)
Fetch the list with given listid from our database, display with template

catalist.views.**register**()
Page for user registration

catalist.views.**signin**()
Sign the user in, given valid credentials.

catalist.views.**signup**()
Sign the user up, given valid credentials and a username the doesn't already exist in our database.

catalist.views.**userlists**(**args*, ***kwargs*)
A page displaying all lists belonging to the user.

1.7 Module contents

copyright

3. 2016 Rachel Wu, Tony Zhang

license lol don't steal this code pls

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

C

catalist.api, 3
catalist.database, 6
catalist.permissions, 11
catalist.views, 11

A

about() (in module catalist.views), 11
acquaintances (catalist.database.User attribute), 9
active (catalist.database.User attribute), 9
add_to_my_lists() (in module catalist.api), 3
admin_unames (in module catalist.permissions), 11
anti_my_lists (catalist.database.User attribute), 9
autocomplete() (in module catalist.api), 3
autocomplete_user() (in module catalist.api), 3

C

Catalist (class in catalist.database), 6
catalist.api (module), 3
catalist.database (module), 6
Catalist.DoesNotExist, 6
Catalist.MultipleObjectsReturned, 6
catalist.permissions (module), 11
catalist.views (module), 11
CatalistEntry (class in catalist.database), 7
CatalistKVP (class in catalist.database), 8
cmp_permission() (in module catalist.permissions), 11
confirmed_at (catalist.database.User attribute), 9
contents (catalist.database.Catalist attribute), 6
contents (catalist.database.CatalistEntry attribute), 8
create_list() (in module catalist.api), 3
created (catalist.database.Catalist attribute), 6
creator (catalist.database.Catalist attribute), 6

D

description (catalist.database.Role attribute), 9
downvoters (catalist.database.CatalistEntry attribute), 8

E

editors (catalist.database.Catalist attribute), 6
email (catalist.database.User attribute), 9
entry_delete() (in module catalist.api), 3
entry_title_save() (in module catalist.api), 3

F

firstname (catalist.database.User attribute), 9
forbidden() (in module catalist.errorviews), 10

G

get_id() (in module catalist.views), 11
get_list_perms() (in module catalist.api), 4
get_pref() (in module catalist.api), 4
getlist() (in module catalist.views), 11

H

handle_invalid_usage() (in module catalist.api), 4
human_readable_time_since() (in module catalist.views), 12

I

id (catalist.database.Catalist attribute), 7
id (catalist.database.Role attribute), 9
id (catalist.database.User attribute), 10
index() (in module catalist.views), 12
internal_server_error() (in module catalist.errorviews), 10
InvalidAPIUsage, 3

K

key (catalist.database.CatalistKVP attribute), 8
key_save() (in module catalist.api), 4
key_val_save() (in module catalist.api), 4
kvp_delete() (in module catalist.api), 4
kvpid (catalist.database.CatalistKVP attribute), 8

L

last_active (catalist.database.User attribute), 10
last_visited (catalist.database.Catalist attribute), 7
lastname (catalist.database.User attribute), 10
list_delete() (in module catalist.api), 4
list_save() (in module catalist.api), 4
list_title_save() (in module catalist.api), 4
listid (catalist.database.Catalist attribute), 7
logged_in() (in module catalist.api), 5
login() (in module catalist.views), 12
logout() (in module catalist.views), 12

M

make_list() (in module catalist.api), 5
method_not_allowed() (in module catalist.errorviews), 10
my_custom_lists (catalist.database.User attribute), 10
my_lists_interact() (in module catalist.api), 5
mylisters (catalist.database.Catalist attribute), 7

N

name (catalist.database.Role attribute), 9

O

objects (catalist.database.Catalist attribute), 7
objects (catalist.database.Role attribute), 9
objects (catalist.database.User attribute), 10
owners (catalist.database.Catalist attribute), 7

P

page_gone() (in module catalist.errorviews), 11
page_not_found() (in module catalist.errorviews), 11
password (catalist.database.User attribute), 10
perm_list (in module catalist.permissions), 11
permissions_forfeit() (in module catalist.api), 5
permissions_get() (in module catalist.api), 5
permissions_set() (in module catalist.api), 5
preferred_theme (catalist.database.User attribute), 10
preview_list() (in module catalist.views), 12
public_level (catalist.database.Catalist attribute), 7
public_level_get() (in module catalist.api), 5
public_level_set() (in module catalist.api), 5

Q

query_cur_perm() (in module catalist.permissions), 11
query_permission() (in module catalist.permissions), 11

R

register() (in module catalist.views), 12
remove_from_my_lists() (in module catalist.api), 5
Role (class in catalist.database), 8
Role.DoesNotExist, 8
Role.MultipleObjectsReturned, 8
roles (catalist.database.User attribute), 10

S

score (catalist.database.CatalistEntry attribute), 8
signin() (in module catalist.views), 12
signup() (in module catalist.views), 12
status_code (catalist.api.InvalidAPIUsage attribute), 3

T

title (catalist.database.Catalist attribute), 7
title (catalist.database.CatalistEntry attribute), 8
to_dict() (catalist.api.InvalidAPIUsage method), 3

U

uid (catalist.database.User attribute), 10
upvoters (catalist.database.CatalistEntry attribute), 8
User (class in catalist.database), 9
User.DoesNotExist, 9
User.MultipleObjectsReturned, 9
userlists() (in module catalist.views), 12

V

value (catalist.database.CatalistKVP attribute), 8
value_save() (in module catalist.api), 5
viewers (catalist.database.Catalist attribute), 7
vote() (in module catalist.api), 5