Nefertari Documentation

Release

Brandicted

Oct 05, 2017

Contents

1 Table of Contents **1**
- 1.1 Base Classes . . . 1
- 1.2 Serializers . . . 3
- 1.3 Fields . . . 3
- 1.4 Changelog . . . 6

CHAPTER 1

Table of Contents

Base Classes

class nefertari_mongodb.documents.**BaseMixin**
Represents mixin class for models.

Attributes:

_auth_fields: String names of fields meant to be displayed to authenticated users.

_public_fields: String names of fields meant to be displayed to non-authenticated users.

_hidden_fields: String names of fields meant to be hidden but editable. _nested_relationships: String names of reference/relationship fields

that should be included in JSON data of an object as full included documents. If reference/relationship field is not present in this list, this field's value in JSON will be an object's ID or list of IDs.

_nesting_depth: Depth of relationship field nesting in JSON. Defaults to 1(one) which makes only one level of relationship nested.

class Q(***query*)
A simple query object, used in a query tree to build up more complex query structures.

BaseMixin.**__weakref__**
list of weak references to the object (if defined)

classmethod BaseMixin.**_delete_many**(*items*, *request=None*)
Delete objects from :items:

BaseMixin.**_is_modified**()
Determine if instance is modified.

For instance to be marked as 'modified', it should:

- Have PK field set (not newly created)

- Have changed fields

BaseMixin.**_to_python_fields**()
Call to_python on non-relation fields.

classmethod BaseMixin.**_update_many**(*items*, *params*, *request=None*)
Update objects from :items:

If :items: is an instance of *mongoengine.queryset.queryset.QuerySet* items.update() is called. Otherwise update is performed per-object.

'on_bulk_update' is called explicitly, because mongoengine does not trigger any signals on QuerySet.update() call.

classmethod BaseMixin.**autogenerate_for**(*model*, *set_to*)
Setup *post_save* event handler.

Event handler is registered for class :model: and creates a new instance of :cls: with a field :set_to: set to an instance on which the event occured.

The handler is set up as class method because mongoengine refuses to call signal handlers if they aren't importable.

classmethod BaseMixin.**expand_with**(*with_cls*, *join_on=None*, *attr_name=None*, *params={}*, *with_params={}*)
Acts like "join" and inserts the with_cls objects in the result as "attr_name".

classmethod BaseMixin.**filter_fields**(*params*)
Filter out fields with invalid names.

classmethod BaseMixin.**filter_objects**(*objects*, *first=False*, ***params*)
Perform query with :params: on instances sequence :objects:

Arguments:

object Sequence of :cls: instances on which query should be run.

params Query parameters to filter :objects:.

classmethod BaseMixin.**get_collection**(***params*)
Params may include '_limit', '_page', '_sort', '_fields'. Returns paginated and sorted query set. Raises JHTTPBadRequest for bad values in params.

classmethod BaseMixin.**get_es_mapping**(*_depth=None*, *types_map=None*)
Generate ES mapping from model schema.

classmethod BaseMixin.**get_null_values**()
Get null values of :cls: fields.

BaseMixin.**get_related_documents**(*nested_only=False*)
Return pairs of (Model, istances) of relationship fields.

Pair contains of two elements:

Model Model class object(s) contained in field.

instances Model class instance(s) contained in field

Parameters **nested_only** – Boolean, defaults to False. When True, return results only contain data for models on which current model and field are nested.

class nefertari_mongodb.documents.**BaseDocument**(**args*, ***values*)

__init__(**args*, ***values*)
Override init to filter out invalid fields from :values:.

Fields are filtered out to make mongoengine less strict when loading objects from database. :internal_fields: are the fields pop'ed from :values: before performing fields presence validation in the original mongoengine init code: https://github.com/MongoEngine/mongoengine/blob/v0.9.0/mongoengine/base/document.py#L41

PS. This issue is fixed in mongoengine master and not released after 0.9.0 yet. https://github.com/MongoEngine/mongoengine/blob/master/mongoengine/base/document.py#L75

clean()
Clean fields which are instances of BaseFieldMixin

classmethod get_field_params(*field_name*)
Get init params of field named :field_name:.

run_backref_hooks()
Runs post-save backref hooks.

Includes backref hooks which are used one time only to sync the backrefs.

save(*request=None*, **arg*, ***kw*)
Force insert document in creation so that unique constraits are respected. This makes each POST to a collection act as a 'create' operation (as opposed to an 'update' for example).

class nefertari_mongodb.documents.**ESBaseDocument**(**args*, ***values*)
Base for document classes which should be indexed by ES.

Serializers

class nefertari_mongodb.serializers.**JSONEncoder**(*skipkeys=False*, *ensure_ascii=True*, *check_circular=True*, *allow_nan=True*, *sort_keys=False*, *indent=None*, *separators=None*, *encoding='utf-8'*, *default=None*)
JSON encoder class to be used in views to encode response.

class nefertari_mongodb.serializers.**ESJSONSerializer**
JSON encoder class used to serialize data before indexing to ES.

Fields

class nefertari_mongodb.fields.**IntegerField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**BigIntegerField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**SmallIntegerField**(**args*, ***kwargs*)
As mongoengine does not provide SmallInt, this is just a copy of *IntegerField*.

class nefertari_mongodb.fields.**BooleanField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**DateField**(**args*, ***kwargs*)
Custom field that stores *datetime.date* instances.

This is basically mongoengine's *DateTimeField* which gets datetime's *.date()* and formats it into a string before storing to mongo.

to_mongo(**args*, ***kwargs*)
Override mongo's DateTimeField value conversion to get date instead of datetime.

class nefertari_mongodb.fields.**DateTimeField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**FloatField**(**args*, ***kwargs*)

validate(*value*)
Override to try to convert string values to floats.

class nefertari_mongodb.fields.**StringField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**TextField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**UnicodeField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**UnicodeTextField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**ChoiceField**(**args*, ***kwargs*)
As mongoengine does not have an explicit ChoiceField, but all mongoengine fields accept *choices* kwarg, we need to define a proxy here. It uses a naive aproach: check the type of first choice and instantiate a field of an appropriate type under the hood. Then translate all the attribute access to the underlying field.

class nefertari_mongodb.fields.**BinaryField**(**args*, ***kwargs*)

translate_kwargs(*kwargs*)
Translate kwargs from one key to another.

Translates: length -> max_bytes

class nefertari_mongodb.fields.**DecimalField**(**args*, ***kwargs*)
This is basically a DecimalField with a fixed name of *precision* kwarg.

translate_kwargs(*kwargs*)
Translate kwargs from one key to another.

Translates: scale -> precision

class nefertari_mongodb.fields.**TimeField**(**args*, ***kwargs*)
Custom field that stores *datetime.date* instances.

class nefertari_mongodb.fields.**PickleField**(**args*, ***kwargs*)
Custom field that stores pickled data as a BinaryField.

Data is pickled when saving to mongo and unpickled when retrieving from mongo. The *pickler* kwarg may be provided that may reference any object with pickle-compatible *dumps* and *loads* methods. Defaults to python's built-in *pickle*.

class nefertari_mongodb.fields.**IntervalField**(**args*, ***kwargs*)
Custom field that stores *datetime.timedelta* instances.

Values are stored as seconds in mongo and loaded by ‘datetime.timedelta(seconds=<value>) when restoring from mongo.

class nefertari_mongodb.fields.**ListField**(**args*, ***kwargs*)
Custom ListField.

The custom part is the validation. Choices are stored in a separate attribute :self.list_choices: and validation checks if the value (which is a sequence) contains anything other than the choices specified in :self.list_choices:.

The original mongoengine ListField validation requires the value to be a sequence but checks for value(sequence) inclusion in choices.

class nefertari_mongodb.fields.**DictField**(**args*, ***kwargs*)

class nefertari_mongodb.fields.**IdField**(**args*, ***kwargs*)
Just a subclass of ObjectIdField that must be used for fields that represent database-specific 'id' field.

class nefertari_mongodb.fields.**ForeignKeyField**(**args*, ***kwargs*)
Field that references another document.

It is not meant to be used inside the mongodb engine. It is added for compatibility with sqla and is not displayed in JSON output.

class nefertari_mongodb.fields.**ReferenceField**(**args*, ***kwargs*)
Field that references another document.

It ISN'T MEANT to be used explicitly by the user. To create a relationship field, use the 'Relationship' constructor function.

When this field is not added to a model's *_nested_relationships*, this field returns an ID of the document that is being referenced. Otherwise the full document is included in JSON response.

This class is used in a *Relationship* function to generate a kind of one-to-one relationship. It is also used to create backreferences.

***reverse_rel_field*: string name of a field on the related document.** Used when generating backreferences so that fields on each side know the name of the field on the other side.

__init__(**args*, ***kwargs*)
Init the field.

Also saves backref kwargs for future creation of the backref.

Expects: *document* or <*_kwargs_prefix*>*document*: mongoengine model name.

__set__(*instance*, *value*)
Custom __set__ method that updates linked relationships.

Updates linked relationship fields if the current field has a *reverse_rel_field* property set. By default this property is only set when the backreference is created.

If value is changed, *instance* is deleted from object it was related to before and is added to object it will be related to now - *value*.

_get_referential_action(*kwargs*)
Determine/translate generic rule name to mongoengine-specific rule.

Custom rule names are used here to make them look SQL-ish and pretty at the same time.

Mongoengine rules are: DO_NOTHING Don't do anything (default) NULLIFY Updates the reference to null CASCADE Deletes the documents associated with the reference DENY Prevent the deletion of the reference object PULL Pull the reference from a ListField of references

_register_addition_hook(*new_object*, *instance*)
Register backref hook to add *instance* to the *new_object's field to which 'instance* became related by the backref.

instance is either added to *new_object* field's collection or *new_object's field, responsible for relationship is set to 'instance*. This depends on type of the field at *new_object*.

instance is not actually used in hook - up-to-date value of *instance* is passed to hook when it is run.

_register_deletion_hook(*old_object*, *instance*)
Register a backref hook to delete the *instance* from the *old_object*'s field to which the *instance* was related before by the backref.

instance is either deleted from the *old_object* field's collection or *old_object*'s field, responsible for relationship is set to None. This depends on type of the field at *old_object*.

instance is not actually used in hook - up-to-date value of *instance* is passed to hook when it is run.

class nefertari_mongodb.fields.**RelationshipField**(**args*, ***kwargs*)

Relationship field meant to be used to create one-to-many relationships.

It ISN'T MEANT to be used by users explicitly. To create a relationship field, use the 'Relationship' constructor function.

It is used in the *Relationship* function to generate one-to-many relationships. Under the hood it is just a ListField containing ReferenceFields.

***reverse_rel_field*: string name of a field on the related document.** Used when generating backreferences so that fields on each side know the name of the field on the other side.

__init__(**args*, ***kwargs*)

Save backref kwargs for future creation of backref.

__set__(*instance*, *value*)

Custom __set__ method that updates linked relationships.

Updates linked relationship fields if current field has *reverse_rel_field* property set. By default this property is only set when a backreference is created.

If the value is changed, *instance* is deleted from the object it was related to before and is added to the object it will be related to now - *value*.

_register_addition_hook(*new_object*, *instance*)

Define and register addition hook.

Hook sets *instance* at *new_object* using *self.reverse_rel_field* as a field which should be set. *instance* is not actually used in hook - up-to-date value of *instance* is passed to hook when it is run.

_register_deletion_hook(*old_object*, *instance*)

Define and register deletion hook.

Hook removes *instance* from *new_object* using *self.reverse_rel_field* as a field from which value should be removed. *instance* is not actually used in hook - up-to-date value of *instance* is passed to hook when it is run.

translate_kwargs(*kwargs*)

For RelationshipField use ReferenceField keys without prefix.

class nefertari_mongodb.fields.**Relationship**

Relationship field generator.

Should be used to generate one-to-many and one-to-one relationships. Provide *uselist* to indicate which kind of relation you expect to get. If *uselist* is True, then RelationshipField is used and a one-to-many is created. Otherwise ReferenceField is used and a one-to-one is created.

This is the place where *ondelete* rules kwargs should be passed. If you switched from the SQLA engine, copy here the same *ondelete* rules you passed to SQLA's *ForeignKeyField*. *ondelete* kwargs may be kept in both fields with no side-effects when switching between the sqla and mongo engines.

Changelog

- #77: Deprecated '_version' field
- : Cosmetic name changes in preparation of engine refactoring

- : Fixed get_field_params() not handling missing fields
- : Partially fixed a performance issue when using backref fields, needs additional work
- : Added '_nesting_depth' property in models to control the level of nesting, default is 1
- : Nested relationships are now indexed in bulk in ElasticSearch
- : Added support for SQLA-like 'onupdate' argument
- : Fixed a bug preventing updates of floatFields via GET tunneling
- : Fixed a bug when using reserved query params with GET tunneling
- : Fixed outdated data in nested relationships of PATCH responses
- : Disabled Elasticsearch indexing of DictField to allow storing arbitrary JSON data
- : Removed 'updated_at' field from engine
- : Fixed bug with Elasticsearch re-indexing of nested relationships
- : Fixed a bug with unicode in StringField
- : Added support for SQLA-like 'onupdate' argument
- : Fixed bug whereby PATCHing relationship field doesn't update all relations
- : Filter-out undefined fields on document load
- : Added python3 support
- : Forward compatibility with nefertari releases
- : Fixed race condition in Elasticsearch indexing
- : Fixed bug with Elasticsearch indexing of nested relationships
- : Fixed password minimum length support by adding before and after validation processors
- : Fixed metaclass fields join
- : Fixed ES mapping error when values of field were all null
- : Relationship indexing

Index

Symbols

__init__() (nefertari_mongodb.documents.BaseDocument method), 2
__init__() (nefertari_mongodb.fields.ReferenceField method), 5
__init__() (nefertari_mongodb.fields.RelationshipField method), 6
__set__() (nefertari_mongodb.fields.ReferenceField method), 5
__set__() (nefertari_mongodb.fields.RelationshipField method), 6
__weakref__ (nefertari_mongodb.documents.BaseMixin attribute), 1
_delete_many() (nefertari_mongodb.documents.BaseMixin class method), 1
_get_referential_action() (nefertari_mongodb.fields.ReferenceField method), 5
_is_modified() (nefertari_mongodb.documents.BaseMixin method), 1
_register_addition_hook() (nefertari_mongodb.fields.ReferenceField method), 5
_register_addition_hook() (nefertari_mongodb.fields.RelationshipField method), 6
_register_deletion_hook() (nefertari_mongodb.fields.ReferenceField method), 5
_register_deletion_hook() (nefertari_mongodb.fields.RelationshipField method), 6
_to_python_fields() (nefertari_mongodb.documents.BaseMixin method), 2
_update_many() (nefertari_mongodb.documents.BaseMixin class method), 2

A

autogenerate_for() (nefertari_mongodb.documents.BaseMixin class method), 2

B

BaseDocument (class in nefertari_mongodb.documents), 2
BaseMixin (class in nefertari_mongodb.documents), 1
BaseMixin.Q (class in nefertari_mongodb.documents), 1
BigIntegerField (class in nefertari_mongodb.fields), 3
BinaryField (class in nefertari_mongodb.fields), 4
BooleanField (class in nefertari_mongodb.fields), 3

C

ChoiceField (class in nefertari_mongodb.fields), 4
clean() (nefertari_mongodb.documents.BaseDocument method), 3

D

DateField (class in nefertari_mongodb.fields), 3
DateTimeField (class in nefertari_mongodb.fields), 4
DecimalField (class in nefertari_mongodb.fields), 4
DictField (class in nefertari_mongodb.fields), 4

E

ESBaseDocument (class in nefertari_mongodb.documents), 3
ESJSONSerializer (class in nefertari_mongodb.serializers), 3
expand_with() (nefertari_mongodb.documents.BaseMixin class method), 2

F

filter_fields() (nefertari_mongodb.documents.BaseMixin class method), 2
filter_objects() (nefertari_mongodb.documents.BaseMixin class method), 2
FloatField (class in nefertari_mongodb.fields), 4

ForeignKeyField (class in nefertari_mongodb.fields), 5

G

get_collection() (nefertari_mongodb.documents.BaseMixin class method), 2
get_es_mapping() (nefertari_mongodb.documents.BaseMixin class method), 2
get_field_params() (nefertari_mongodb.documents.BaseDocument class method), 3
get_null_values() (nefertari_mongodb.documents.BaseMixin class method), 2
get_related_documents() (nefertari_mongodb.documents.BaseMixin method), 2

I

IdField (class in nefertari_mongodb.fields), 5
IntegerField (class in nefertari_mongodb.fields), 3
IntervalField (class in nefertari_mongodb.fields), 4

J

JSONEncoder (class in nefertari_mongodb.serializers), 3

L

ListField (class in nefertari_mongodb.fields), 4

P

PickleField (class in nefertari_mongodb.fields), 4

R

ReferenceField (class in nefertari_mongodb.fields), 5
Relationship (class in nefertari_mongodb.fields), 6
RelationshipField (class in nefertari_mongodb.fields), 6
run_backref_hooks() (nefertari_mongodb.documents.BaseDocument method), 3

S

save() (nefertari_mongodb.documents.BaseDocument method), 3
SmallIntegerField (class in nefertari_mongodb.fields), 3
StringField (class in nefertari_mongodb.fields), 4

T

TextField (class in nefertari_mongodb.fields), 4
TimeField (class in nefertari_mongodb.fields), 4
to_mongo() (nefertari_mongodb.fields.DateField method), 3
translate_kwargs() (nefertari_mongodb.fields.BinaryField method), 4
translate_kwargs() (nefertari_mongodb.fields.DecimalField method), 4
translate_kwargs() (nefertari_mongodb.fields.RelationshipField method), 6

U

UnicodeField (class in nefertari_mongodb.fields), 4
UnicodeTextField (class in nefertari_mongodb.fields), 4

V

validate() (nefertari_mongodb.fields.FloatField method), 4