avocato Documentation

Release 0.0.1

Tomaz Sifrer

Jan 11, 2019

Contents:

1 Installation **3**

2 Documentation **5**

3 Example **7**

3.1 API Reference 7

3.2 Vendors 10

4 Indices and tables **13**

avocato is a simple and fast ORM/framework-agnostic object serialization library for converting complex objects to and from simple Python datatypes.

Don't be scared if you're using an ORM/framework. It can easily be adapted to be used with any ORM/framework of your liking. Currently it supports Django ORM and peewee.

This library is heavily influenced by serpy.

CHAPTER 1

Installation

```
$ pip install avocato
```

CHAPTER 2

Documentation

Find documentation at avocato.rtfd.io

CHAPTER 3

Example

```
import avocato

class Bar(object):
    patrick = 'star'


class Foo(object):
    over = 9000
    spongebob = 'squarepants'
    bar = Bar()


class BarSerializer(avocato.Serializer):
    patrick = avocato.StrField()


class FooSerializer(avocato.Serializer):
    over = avocato.IntField()
    spongebob = avocato.StrField()
    bar = BarSerializer()


foo = Foo()
FooSerializer(foo).data
# {'over': 9000, 'spongebob': 'squarepants', 'bar': {'patrick': 'star'}}
```

3.1 API Reference

3.1.1 Serializers

class avocato.**Serializer**(*instance=None*, *many=False*, *data=None*, ***kwargs*)

Base class for custom object serializers.

`Serializer` can be used as a Field to create nested schemas. A serializer is defined by subclassing Serializer and adding each Field as a class variable:

Example:

```
class MyObject(object):
    bar = 2
    baz = 'hello'

class FooSerializer(Serializer):
    bar = IntField()
    baz = StrField()

obj = MyObject()
FooSerializer(obj).data
# {'bar': 2, 'baz': 'hello'}
```

data
Get the serialized data from the *Serializer*.

The data will be cached for future accesses.

is_valid()
Checks wether data passes validation.

Returns True if all validations were successful on all fields, otherwise returns False.

to_instance(*instance_class=None*)
Populates an instance with data

If doesn't exists, it creates a new one and populates it. If it already exists, it updates fields with new data.

class avocato.**DictSerializer**(*instance=None*, *many=False*, *data=None*, ***kwargs*)
Base class for custom `dict` serializers.

Example:

```
class FooSerializer(DictSerializer):
    bar = IntField()
    baz = StrField()

obj = {'bar': 2, 'baz': 'hello'}
FooSerializer(obj).data
# {'bar': 2, 'baz': 'hello'}
```

3.1.2 Fields

class avocato.**Field**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Field handles converting between primitive values and internal datatypes. It also deals with validating input values.

Parameters

- **attr** (*str*) – The attribute to get on the object. If this is not supplied, the name this field was assigned to on the serializer will be used.
- **label** (*str*) – A label to use as the name of the serialized field instead of using the attribute name of the field.
- **required** (*bool*) – Whether the field is required.

- **validators** (*list*) – List of validators to run when calling `.is_valid()` method on the serializer.
- **call** (*bool*) – Whether the value should be called after it is retrieved from the object. Useful if an object has a method to be serialized.
- **is_create_field** (*bool*) – Whether the field is used to populate the instance when creating a new object via `to_instance` method on the serializer.
- **call** – Whether the field is used to populate the instance when updating an object via `to_instance` method on the serializer.

class avocato.**StrField**(***kwargs*)
Converts input value to string.

Parameters

- **max_length** (*int*) – Maximum lenght of the string. If present, adds a validator for max lenght which will be run when `is_valid` method on the serializer is called.
- **min_length** (*int*) – Minimum lenght of the string. If present, adds a validator for min lenght which will be run when `is_valid` method on the serializer is called.
- **choices** (*list*) – Available choices. If present, adds a validator that checks if value is present in choices and will be run when `is_valid` method on the serializer is called.

class avocato.**EmailField**(***kwargs*)
Converts input value to email.

class avocato.**IntField**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Converts input value to integer.

class avocato.**FloatField**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Converts input value to float.

class avocato.**BoolField**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Converts input value to bool.

class avocato.**DecimalField**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Converts input value to string, so it accurately shows decimal numbers.

class avocato.**DateTimeField**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Converts input value to ISO format date.

class avocato.**DictField**(*attr=None*, *label=None*, *required=True*, *validators=None*, *call=False*, *is_create_field=True*, *is_update_field=True*)
Converts input value to dict.

class avocato.**MethodField**(*method=None*, ***kwargs*)
Calls a method on the *Serializer* to get the value.

3.1.3 Validators

class avocato.**Validator**
Base class for validators.

class avocato.**Required**(*message=None*)
Validates if value is set.

Raises exception if value is empty or None

class avocato.**Email**(*message=None*)
Validates if value is in valid email format

class avocato.**Length**(*min_length=None*, *max_length=None*, *message=None*, *equal=None*)
Validates if value is correct size.

class avocato.**OneOf**(*choices*, *message=None*)
Validates if the value is one of the choices.

class avocato.**OneOfType**(*choices*, *message=None*)
Validates if value type is one of the choices.

3.1.4 Exceptions

class avocato.**AvocatoError**
Base avocato exception.

class avocato.**AvocatoValidationError**(*message*, *field_names=None*, *data=None*, *valid_data=None*, ***kwargs*)
Exception used for validating values.

3.2 Vendors

3.2.1 Django

Serializer for easily converting to and from simple Python datatypes to Django Models.

class avocato.vendors.django.**DjangoModelSerializer**(*instance=None*, *many=False*, *data=None*, ***kwargs*)
Class for converting to and from simple Python datatypes to Django Models.

Example:

```
class ModelFoo(django.db.models.Model):
    bar = models.IntegerField()
    baz = models.CharField()

class FooSerializer(DjangoModelSerializer):
    class Meta:
        model = ModelFoo
        fields =['bar', 'baz']

obj = ModelFoo(bar=2, baz='hello')
FooSerializer(obj).data
# {'bar': 2, 'baz': 'hello'}
```

create(*data*)
Override this function if you want to do something custom when creating an object.

save()
Saves the instance with the help of Django models `.save()` method.

update(*data*)
Override this function if you want to do something custom when updating an object.

3.2.2 peewee

Serializer for easily converting to and from simple Python datatypes to peewee models.

class avocato.vendors.peewee.**PeeWeeModelSerializer**(*instance=None*, *many=False*, *data=None*, ***kwargs*)
Class for converting to and from simple Python datatypes to peewee models.

Example:

```
class ModelFoo(peewee.Model):
    bar = peewee.IntegerField()
    baz = peewee.CharField()

class FooSerializer(PeeweeModelSerializer):
    class Meta:
        model = ModelFoo
        fields =['bar', 'baz']

obj = ModelFoo(bar=2, baz='hello')
FooSerializer(obj).data
# {'bar': 2, 'baz': 'hello'}
```

create(*data*)
Override this function if you want to do something custom when creating an object.

save()
Saves the instance with the help of peewee models `.save()` method.

update(*data*)
Override this function if you want to do something custom when updating an object.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index

A

AvocatoError (class in avocato), 10
AvocatoValidationError (class in avocato), 10

B

BoolField (class in avocato), 9

C

create() (avocato.vendors.django.DjangoModelSerializer method), 10
create() (avocato.vendors.peewee.PeeWeeModelSerializer method), 11

D

data (avocato.Serializer attribute), 8
DateTimeField (class in avocato), 9
DecimalField (class in avocato), 9
DictField (class in avocato), 9
DictSerializer (class in avocato), 8
DjangoModelSerializer (class in avocato.vendors.django), 10

E

Email (class in avocato), 10
EmailField (class in avocato), 9

F

Field (class in avocato), 8
FloatField (class in avocato), 9

I

IntField (class in avocato), 9
is_valid() (avocato.Serializer method), 8

L

Length (class in avocato), 10

M

MethodField (class in avocato), 9

O

OneOf (class in avocato), 10
OneOfType (class in avocato), 10

P

PeeWeeModelSerializer (class in avocato.vendors.peewee), 11

R

Required (class in avocato), 9

S

save() (avocato.vendors.django.DjangoModelSerializer method), 10
save() (avocato.vendors.peewee.PeeWeeModelSerializer method), 11
Serializer (class in avocato), 7
StrField (class in avocato), 9

T

to_instance() (avocato.Serializer method), 8

U

update() (avocato.vendors.django.DjangoModelSerializer method), 10
update() (avocato.vendors.peewee.PeeWeeModelSerializer method), 11

V

Validator (class in avocato), 9