
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 avocado
```


CHAPTER 2

Documentation

Find documentation at avocato.rtd.io


```
import avocado

class Bar(object):
    patrick = 'star'

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

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

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

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

3.1 API Reference

3.1.1 Serializers

class `avocado.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 whether 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 exist, it creates a new one and populates it. If it already exists, it updates fields with new data.

class `avocado.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 `avocado.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 length of the string. If present, adds a validator for max length which will be run when `is_valid` method on the serializer is called.
- **min_length** (*int*) – Minimum length of the string. If present, adds a validator for min length 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.BooleanField` (*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 `avocado.Required` (*message=None*)
 Validates if value is set.

Raises exception if value is empty or None

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

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

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

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

3.1.4 Exceptions

class `avocado.AvocadoError`
 Base avocado exception.

class `avocado.AvocadoValidationError` (*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 `avocado.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 avocado.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`

A

AvocadoError (class in avocado), 10
AvocadoValidationError (class in avocado), 10

B

BoolField (class in avocado), 9

C

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

D

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

E

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

F

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

I

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

L

Length (class in avocado), 10

M

MethodField (class in avocado), 9

O

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

P

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

R

Required (class in avocado), 9

S

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

T

to_instance() (avocado.Serializer method), 8

U

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

V

Validator (class in avocado), 9