sanic-transmute Documentation

Release 0.1

Yun Xu

Nov 21, 2018

Contents

Easily document your Sanic API with Swagger UI, Plus param validation and model serialization.

CHAPTER 1

What is sanic-transmute ?

A transmute framework for sanic. This framework provides:

- declarative generation of http handler interfaces by parsing function annotations
- validation and serialization to and from a variety of content types (e.g. json or yaml).
- validation and serialization to and from native python objects, using attrs and schematics.
- autodocumentation of all handlers generated this way, via swagger.

CHAPTER 2

Quick start

Let’s get started.

Find Examples here:

- example with attrs model.
- example with schematic model.

Simple example with schematics model.

```
from sanic import Sanic, Blueprint
from sanic.response import json
from sanic_transmute import describe, add_route, add_swagger, APIException
from sanic.exceptions import ServerError
from schematics.models import Model
from schematics.types import IntType


class User(Model):
    points = IntType()


app = Sanic()
bp = Blueprint("test_blueprints", url_prefix="/blueprint")


@describe(paths="/api/v1/user/{user}/", methods="GET")
async def test_transmute(request, user: str, env: str=None, group: [str]=None):
    """
    API Description: Transmute Get. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    return {
        "user": user,
        "env": env,
        "group": group,
```

(continues on next page)

(continued from previous page)

```
    }


@describe(paths="/killme")
async def handle_exception(request) -> User:
    """
    API Description: Handle exception. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise ServerError("Something bad happened", status_code=500)


@describe(paths="/api/v1/user/missing")
async def handle_api_exception(request) -> User:
    """
    API Description: Handle APIException. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise APIException("Something bad happened", code=404)


@describe(paths="/multiply")
async def get_blueprint_params(request, left: int, right: int) -> str:
    """
    API Description: Multiply, left * right. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    res = left * right
    return "{left}*{right}={res}".format(left=left, right=right, res=res)


if __name__ == "__main__":
    add_route(app, test_transmute)
    add_route(app, handle_exception)
    add_route(app, handle_api_exception)
    # register blueprints
    add_route(bp, get_blueprint_params)
    app.blueprint(bp)
    # add swagger
    add_swagger(app, "/api/v1/swagger.json", "/api/v1/")
    app.run(host="0.0.0.0", port=8000)
```

Simple example with attrs model.

```
from sanic import Sanic, Blueprint
from sanic.response import json
from sanic_transmute import describe, add_route, add_swagger, APIException
from sanic.exceptions import ServerError
import attr


@attr.s
class User:
    points = attr.ib(type=int)


app = Sanic()
```

(continues on next page)

(continued from previous page)

```
bp = Blueprint("test_blueprints", url_prefix="/blueprint")


@describe(paths="/api/v1/user/{user}/", methods="GET")
async def test_transmute_get(request, user: str, env: str=None, group: [str]=None):
    """
    API Description: Transmute Get. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    return {
        "user": user,
        "env": env,
        "group": group,
    }


@describe(paths="/api/v1/user/", methods="POST")
async def test_transmute_post(request, user: User) -> User:
    """
    API Description: Transmute Post. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    return user


@describe(paths="/killme")
async def handle_exception(request) -> User:
    """
    API Description: Handle exception. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise ServerError("Something bad happened", status_code=500)


@describe(paths="/api/v1/user/missing")
async def handle_api_exception(request) -> User:
    """
    API Description: Handle APIException. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise APIException("Something bad happened", code=404)


@describe(paths="/multiply")
async def get_blueprint_params(request, left: int, right: int) -> str:
    """
    API Description: Multiply, left * right. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    res = left * right
    return "{left}*{right}={res}".format(left=left, right=right, res=res)


if __name__ == "__main__":
    add_route(app, test_transmute_get)
    add_route(app, test_transmute_post)
    add_route(app, handle_exception)
```

(continues on next page)

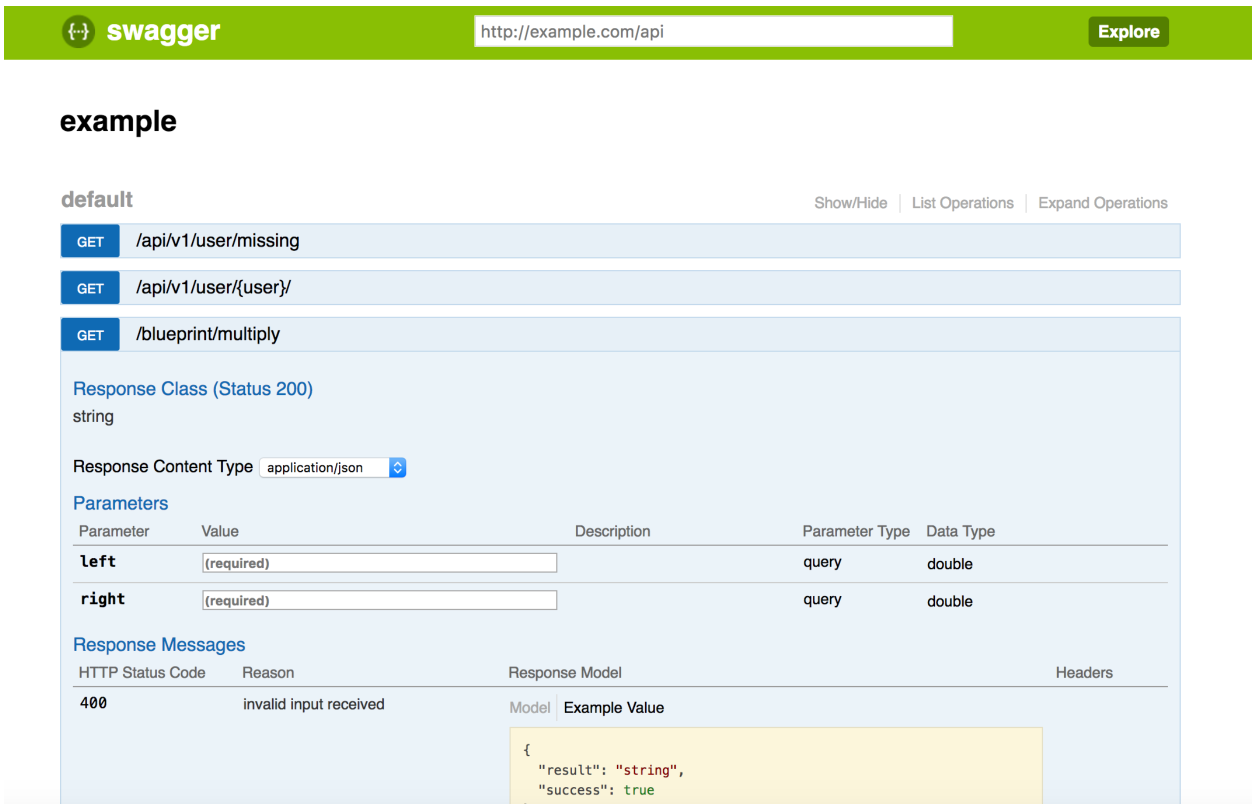
(continued from previous page)

```
    add_route(app, handle_api_exception)
    # register blueprints
    add_route(bp, get_blueprint_params)
    app.blueprint(bp)
    # add swagger
    add_swagger(app, "/api/v1/swagger.json", "/api/v1/")
    app.run(host="0.0.0.0", port=8000)
```

CHAPTER 3

Swagger Integration

You can get Swagger UI for free.



example

default

GET /api/v1/user/missing

GET /api/v1/user/{user}/

GET /blueprint/multiply

Response Class (Status 200)

string

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
left	(required)		query	double
right	(required)		query	double

Response Messages

HTTP Status Code	Reason	Response Model	Headers
400	invalid input received	Model Example Value	

```
{
  "result": "string",
  "success": true
```

Contents:

3.1 Installation

3.1.1 Installing it globally

You can install sanic-transmute globally with any Python package manager:

```
pip install sanic-transmute
```

3.2 Example

Find Examples here:

- example with attrs model.
- example with schematic model.

Simple example with schematics model.

```
from sanic import Sanic, Blueprint
from sanic.response import json
from sanic_transmute import describe, add_route, add_swagger, APIException
from sanic.exceptions import ServerError
from schematics.models import Model
from schematics.types import IntType


class User(Model):
    points = IntType()


app = Sanic()
bp = Blueprint("test_blueprints", url_prefix="/blueprint")


@describe(paths="/api/v1/user/{user}/", methods="GET")
async def test_transmute(request, user: str, env: str=None, group: [str]=None):
    """
    API Description: Transmute Get. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    return {
        "user": user,
        "env": env,
        "group": group,
    }


@describe(paths="/killme")
async def handle_exception(request) -> User:
    """
    API Description: Handle exception. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise ServerError("Something bad happened", status_code=500)
```

(continues on next page)

(continued from previous page)

```
@describe(paths="/api/v1/user/missing")
async def handle_api_exception(request) -> User:
    """
    API Description: Handle APIException. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise APIException("Something bad happened", code=404)


@describe(paths="/multiply")
async def get_blueprint_params(request, left: int, right: int) -> str:
    """
    API Description: Multiply, left * right. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    res = left * right
    return "{left}*{right}={res}".format(left=left, right=right, res=res)


if __name__ == "__main__":
    add_route(app, test_transmute)
    add_route(app, handle_exception)
    add_route(app, handle_api_exception)
    # register blueprints
    add_route(bp, get_blueprint_params)
    app.blueprint(bp)
    # add swagger
    add_swagger(app, "/api/v1/swagger.json", "/api/v1/")
    app.run(host="0.0.0.0", port=8000)
```

Simple example with attrs model.

```
from sanic import Sanic, Blueprint
from sanic.response import json
from sanic_transmute import describe, add_route, add_swagger, APIException
from sanic.exceptions import ServerError
import attr


@attr.s
class User:
    points = attr.ib(type=int)


app = Sanic()
bp = Blueprint("test_blueprints", url_prefix="/blueprint")


@describe(paths="/api/v1/user/{user}/", methods="GET")
async def test_transmute_get(request, user: str, env: str=None, group: [str]=None):
    """
    API Description: Transmute Get. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    return {
        "user": user,
```

(continues on next page)

(continued from previous page)

```
        "env": env,
        "group": group,
    }


@describe(paths="/api/v1/user/", methods="POST")
async def test_transmute_post(request, user: User) -> User:
    """
    API Description: Transmute Post. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    return user


@describe(paths="/killme")
async def handle_exception(request) -> User:
    """
    API Description: Handle exception. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise ServerError("Something bad happened", status_code=500)


@describe(paths="/api/v1/user/missing")
async def handle_api_exception(request) -> User:
    """
    API Description: Handle APIException. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    raise APIException("Something bad happened", code=404)


@describe(paths="/multiply")
async def get_blueprint_params(request, left: int, right: int) -> str:
    """
    API Description: Multiply, left * right. This will show in the swagger page␣
↪(localhost:8000/api/v1/).
    """
    res = left * right
    return "{left}*{right}={res}".format(left=left, right=right, res=res)


if __name__ == "__main__":
    add_route(app, test_transmute_get)
    add_route(app, test_transmute_post)
    add_route(app, handle_exception)
    add_route(app, handle_api_exception)
    # register blueprints
    add_route(bp, get_blueprint_params)
    app.blueprint(bp)
    # add swagger
    add_swagger(app, "/api/v1/swagger.json", "/api/v1/")
    app.run(host="0.0.0.0", port=8000)
```

3.3 Routes

3.3.1 Example

Adding routes follows the standard transmute pattern, with a decorator converting a function to an aiohttp route:

```python
from sanic_transmute import describe, add_route
from sanic import Sanic

app = Sanic()

# define a GET endpoint, taking a query parameter integers left and right,
# which must be integers.
@describe(paths="/{name}")
async def multiply(request, name: str, left: int, right: int) -> int:
    return left + right

# append to your route later
add_route(app, multiply)
```

the sanic request argument is supported: it will be passed into any function that has ‘request’ in it’s function signature.

see transmute-core:function for more information on customizing transmute routes.

3.3.2 API Documentation

`sanic_transmute.`**`describe`**`(`*`**kwargs`*`)`

describe is a decorator to customize the rest API that transmute generates, such as choosing certain arguments to be query parameters or body parameters, or a different method.

Parameters

- **paths** (*list (str)*) – the path(s) for the handler to represent (using swagger’s syntax for a path)
- **methods** (*list (str)*) – the methods this function should respond to. if non is set, transmute defaults to a GET.
- **query_parameters** (*list (str)*) – the names of arguments that should be query parameters. By default, all arguments are query_or path parameters for a GET request.
- **body_parameters** (*List [str] or str*) – the names of arguments that should be body parameters. By default, all arguments are either body or path parameters for a non-GET request.

  in the case of a single string, the whole body is validated against a single object.
- **header_parameters** (*list (str)*) – the arguments that should be passed into the header.
- **path_parameters** (*list (str)*) – the arguments that are specified by the path. By default, arguments that are found in the path are used first before the query_parameters and body_parameters.
- **parameter_descriptions** (*list (str)*) – descriptions for each parameter, keyed by attribute name. this will appear in the swagger documentation.

3.4 Serialization

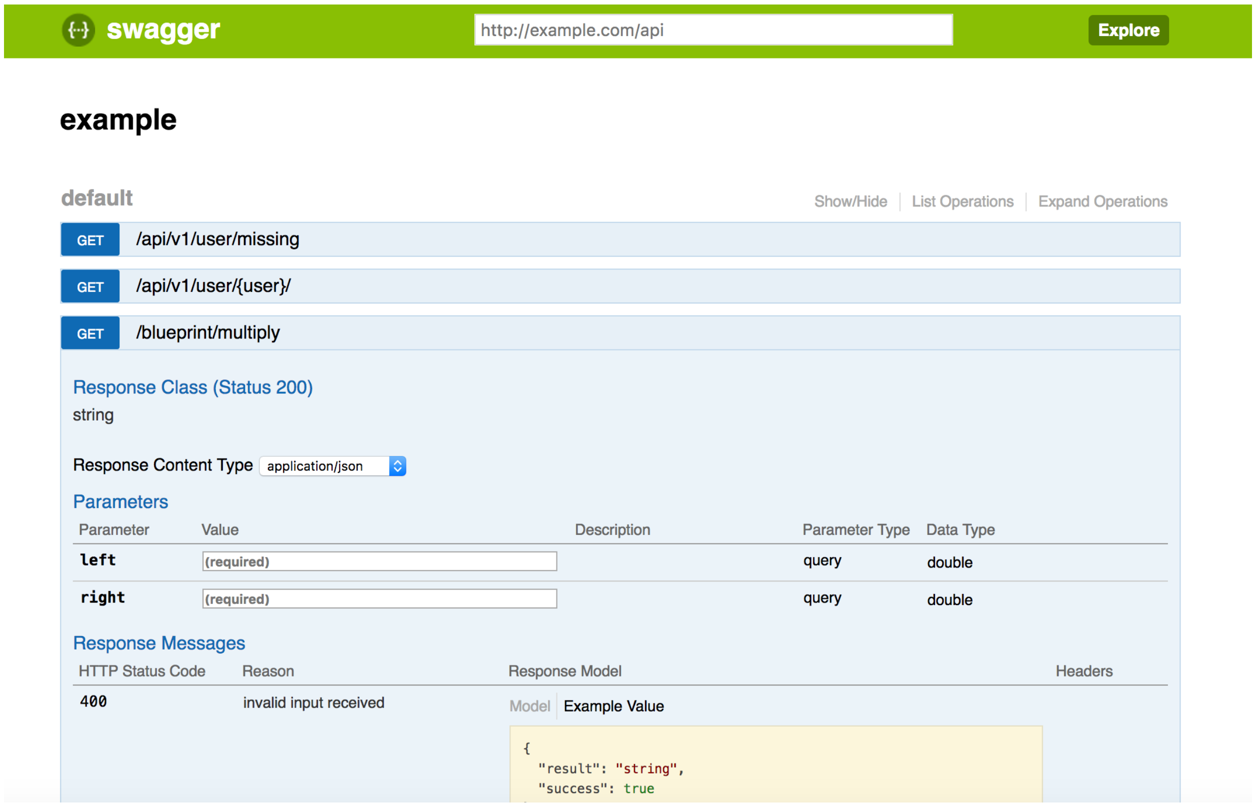
See serialization in transmute-core.

3.5 Autodocumentation

You can use add_swagger(app, json_path, html_path) to add swagger documentation for all transmute routes.

```
sanic_transmute.add_swagger(app, "/api/v1/swagger.json", "/ap1/v1")
```

The swagger page looks like,



example

default

GET /api/v1/user/missing

GET /api/v1/user/{user}/

GET /blueprint/multiply

Response Class (Status 200)
string

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
left	(required)		query	double
right	(required)		query	double

Response Messages

HTTP Status Code	Reason	Response Model	Headers
400	invalid input received		

```
{
  "result": "string",
  "success": true
```

3.5.1 API Reference

`sanic_transmute.add_swagger`(*app*, *json_route*, *html_route*)
a convenience method for both adding a swagger.json route, as well as adding a page showing the html documentation

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index