Flask-Journey Documentation

Release 0.1.4

Robert Wikman

Oct 10, 2018

Contents

1 Installation **3**

2 Journey Usage **5**

3 The route decorator **7**

4 Blueprints **9**
- 4.1 Bundling blueprints . . . 9
- 4.2 Importing bundles . . . 9

5 API Documentation **11**
- 5.1 Journey API . . . 11
- 5.2 Route decorator . . . 12
- 5.3 BlueprintBundle API . . . 13
- 5.4 Exceptions . . . 13

Python Module Index **15**

The two core components of Journey, `route` and `BlueprintBundle`, are not dependent on each other, however, there might be code added in the future that will enable them to integrate.

This, and the fact that they operate in the same field was the motivation for adding both to this extension.

CHAPTER 1

Installation

Use pip to install the extension:

```
$ pip install flask-journey
```

CHAPTER 2

Journey Usage

This step is only necessary if you plan on using the BlueprintBundle

The extension is managed through a `Journey` instance. If utilizing application factories, then you probably want to go the init_app() route:

```
from flask import Flask
from flask_journey import Journey

from .bundles import bundle1, bundle2

app = Flask(__name__)
journey = Journey()
journey.attach_bundle(bundle1)
journey.attach_bundle(bundle2)
journey.init_app(app)
```

You may also set up `Journey` directly, passing a list of bundles its constructor:

```
app = Flask(__name__)
journey = Journey(app, bundles=[bundle1, bundle2])
```

CHAPTER 3

The route decorator

The `route` component, as mentioned previously, is not dependent on the Journey blueprint manager. However, functions decorated with `flask_journey.route` can of course, just as `flask.Blueprint.route`, be added to your app with the help of Journey.

Marshmallow compatible schemas:

```
# file: api/users/schemas.py

from marshmallow import Schema, fields, validate

class QuerySchema(Schema):
    first_name = fields.String(required=False)
    last_name = fields.String(required=False)


class UserSchema(Schema):
    id = fields.Integer(required=True)
    first_name = fields.String(required=True)
    last_name = fields.String(required=True)
    user_name = fields.String(required=True)


users = UserSchema(many=True)
user = UserSchema()
query = QuerySchema()
```

. . . with the flask_journey.route decorator enables simple (de)serialization and validation:

```
# api/users/controllers.py

from flask import Blueprint
from flask_journey import route

from .services import create_user, get_user, update_user
```

(continues on next page)

(continued from previous page)

```
from .schemas import user, users, query

bp = Blueprint('users', __name__)

@route(bp, '/', methods=['GET'], _query=query, marshal_with=users)
def get_many(_query):
    return get_users(_query.data)


@route(bp, '/', methods=['POST'], _body=user, marshal_with=user)
def create(_body):
    return create_user(_body.data)


@route(bp, '/<user_id>', methods=['PUT'], _body=user, marshal_with=user)
def update(user_id, _body):
    return update_user(user_id, _body.data)
```

CHAPTER 4

Blueprints

4.1 Bundling blueprints

There are various benefits of using the Journey BlueprintBundle, and in most cases just one BlueprintBundle is enough.

- It can be used to easily segregate your blueprint registration code from the other parts of your application.
- It helps you group blueprints in a logical manner.
- It enables you to utilize the Journey API (currently only for blueprint bundle registration and listing routes)

```
# file: api/bundles.py

from flask_journey import BlueprintBundle

from .users import bp as users
from .groups import bp as groups
from .companies import bp as companies
from .stuff import bp as stuff

v1 = BlueprintBundle(path='/api/v1', description="API v1, stable")
v1.attach_bp(users, description='Users CRUD')
v1.attach_bp(groups)
v1.attach_bp(companies, description='Companies API')

other = BlueprintBundle(path='/other')
other.attach_bp(stuff)
```

4.2 Importing bundles

Importing and registering bundles (along with blueprints) is easy as pie:

```
# file: api/__init__.py

from flask import Flask
from .bundles import v1, other

app = Flask(__name__)
journey = Journey()
journey.attach_bundle(v1)
journey.attach_bundle(other)
journey.init_app(app)
```

CHAPTER 5

API Documentation

5.1 Journey API

class flask_journey.**Journey**(*app=None*, *bundles=None*)

Central controller class. Exposes an API for managing blueprints and listing routes

Parameters **app** – App to pass directly to Journey

Raises

- InvalidBundlesType if passed bundles is not of type list

attach_bundle(*bundle*)

Attaches a bundle object

Parameters **bundle** – *flask_journey.BlueprintBundle* object

Raises

- IncompatibleBundle if the bundle is not of type *BlueprintBundle*
- ConflictingPath if a bundle already exists at bundle.path
- MissingBlueprints if the bundle doesn't contain any blueprints

static get_blueprint_routes(*app*, *base_path*)

Returns detailed information about registered blueprint routes matching the *BlueprintBundle* path

Parameters

- **app** – App instance to obtain rules from
- **base_path** – Base path to return detailed route info for

Returns List of route detail dicts

static get_bp_path(*bp*)

Returns url_prefix if set, otherwise bp.name prefixed with a slash.

Parameters **bp** – flask.Blueprint object

Returns blueprint name

init_app(*app*)
Initializes Journey extension

Parameters **app** – App passed from constructor or directly to init_app

Raises

- NoBundlesAttached if no bundles has been attached attached

routes_detailed
Returns a detailed list of bundles along with blueprints and routes

Returns List of blueprint routes

routes_simple
Returns simple info about registered blueprints

Returns Tuple containing endpoint, path and allowed methods for each route

5.2 Route decorator

flask_journey.utils.**route**(*bp*, **args*, ***kwargs*)
Journey route decorator

Enables simple serialization, deserialization and validation of Flask routes with the help of Marshmallow.

Parameters

- **bp** – `flask.Blueprint` object
- **args** – args to pass along to *Blueprint.route*
- **kwargs** –
  - **strict_slashes** Enable / disable strict slashes (default False)
  - **validate** Enable / disable body/query validation (default True)
  - **_query** Unmarshal Query string into this schema
  - **_body** Unmarshal JSON body into this schema
  - **marshal_with** Serialize the output with this schema

Raises

- ValidationError if the query parameters or JSON body fails validation

flask_journey.utils.**sanitize_path**(*path*)
Performs sanitation of the path after validating

Parameters **path** – path to sanitize

Returns path

Raises

- InvalidPath if the path doesn't start with a slash

5.3 BlueprintBundle API

class flask_journey.**BlueprintBundle**(*path='/'*, *description=''*)
Creates a BlueprintBundle at the path specified

Parameters **path** – blueprint base path

attach_bp(*bp*, *description=''*)
Attaches a flask.Blueprint to the bundle

Parameters

- **bp** – `flask.Blueprint` object
- **description** – Optional description string

Raises

- InvalidBlueprint if the Blueprint is not of type *flask.Blueprint*

5.4 Exceptions

exception flask_journey.exceptions.**ConflictingPath**

exception flask_journey.exceptions.**IncompatibleBundle**

exception flask_journey.exceptions.**IncompatibleSchema**

exception flask_journey.exceptions.**InvalidBlueprint**

exception flask_journey.exceptions.**InvalidBundlesType**

exception flask_journey.exceptions.**InvalidPath**

exception flask_journey.exceptions.**MissingBlueprints**

exception flask_journey.exceptions.**NoBundlesAttached**

Python Module Index

f

flask_journey, **??**
flask_journey.exceptions, 13
flask_journey.utils, 12

Index

A

attach_bp() (flask_journey.BlueprintBundle method), 13
attach_bundle() (flask_journey.Journey method), 11

B

BlueprintBundle (class in flask_journey), 13

C

ConflictingPath, 13

F

flask_journey (module), 1
flask_journey.exceptions (module), 13
flask_journey.utils (module), 12

G

get_blueprint_routes() (flask_journey.Journey static method), 11
get_bp_path() (flask_journey.Journey static method), 11

I

IncompatibleBundle, 13
IncompatibleSchema, 13
init_app() (flask_journey.Journey method), 12
InvalidBlueprint, 13
InvalidBundlesType, 13
InvalidPath, 13

J

Journey (class in flask_journey), 11

M

MissingBlueprints, 13

N

NoBundlesAttached, 13

R

route() (in module flask_journey.utils), 12
routes_detailed (flask_journey.Journey attribute), 12
routes_simple (flask_journey.Journey attribute), 12

S

sanitize_path() (in module flask_journey.utils), 12