Flask-OpenAPI

Release

Oct 07, 2017

Contents

CHAPTER 1

flask_openapi

flask_openapi.extension

This module contains the actual Flask extension.

class `flask_openapi.extension.`**`OpenAPI`**(*app=None*)

The Flask extension that handles all magic.

Parameters **`app`** (*`flask.Flask`*) – The Flask app to apply this extension to.

`add_definition`(*definition*, *name=None*)

Add a new named definition.

Parameters

- **`definition`** – A `dict` describing a Swagger schemaObject. This may also be a `str` or `pathlib.Path` referring to a file to read.
- **`name`** (*`str`*) – A name for the schema. If this is not specified, the title of the schema definition will be used.

`add_response`(*name*, *response*)

Define a possible response to be reused accross operations.

Parameters

- **`name`** (*`str`*) – The name of the response. See Swagger rdName.
- **`response`** (*`dict`*) – The response to add. See Swagger responseObject.

`base_path`

`str` – The relative URL prefix for all API calls.

See Swagger swaggerBasePath for details.

`definitions`

`dict` – The top level Swagger definitionsObject.

`deprecated`(*fn*)

Mark an operation as deprecated.

This will be exposed through the OpenAPI operation object. Additionally a warning will be emitted when the API is used. This can be configured using the `OPENAPI_WARN_DEPRECATED` configuration option. This must be one of `warn` or `log`.

See Swagger operationDeprecated.

host
str – The server host name.

This is only returned if `show_host` is `True`.

See Swagger swaggerHost for details.

info
dict – The top level Swagger infoObject.

init_app(*app*)
Initialize the OpenAPI instance for the given app.

This should be used for a deferred initialization, supporting the Flask factory pattern.

Parameters **app** (*flask.Flask*) – The Flask app to apply this extension to.

paths
dict – The top level Swagger pathsObject.

response(*status_code*, *response*)
Describe a possible response for a Flask handler.

Parameters

- **status_code** – The status code for which the response is described. str, int or http.HTTPStatus are accepted. The result will be exposed as a string. See Swagger responsesCode.
- **response** – A description of the response object. This may be a dict describing a response or a string referring to a response added using *add_response*. See Swagger responseObject.

responses
dict – The top level Swagger responsesObject.

schema(*schema*)
A decorator to validate a request using a Swagger schemaObject.

Parameters **schema** – Either a dict to use as a schema directly or a str referring to a named schema. (See *add_definition*.)

schemes
list – The supported schemes for all API calls.

See Swagger swaggerSchemes for details.

swagger
dict – The top level Swagger swaggerObject.

tag(**tags*)
Tag an operation using one or more tags.

These tags are exposed through the OpenAPI operation object.

Parameters ***tags** (*str*) – The tags to apply to the operation.

tags
dict – A list of tag descriptions.

See Swagger tagObject.

validatorgetter(*fn*)
Mark a function as a getter function to get a validator.

The function will be called with the JSON schema as a dict.

exception flask_openapi.extension.**UnknownDefinitionError**(*name*)
Raised when trying to get a definition which doesn't exist.

Parameters **name** (*str*) – The definition that could not be found.

exception flask_openapi.extension.**UnnamedDefinitionError**(*definition*)
Raised when trying to add a definition which has no title.

Parameters **definition** (*dict*) – The unnamed JSON schema definition.

flask_openapi.utils

This module contains some utility functions.

flask_openapi.utils.**add_optional**(*data*, *key*, *value*)
Add a value to the data dict, but only if the value is not None.

Parameters

- **data** (*dict*) – The dict to add the value to.
- **key** (*str*) – The key to assign the value to in the data dict.
- **value** – The value to assign if it's not None.

flask_openapi.utils.**parse_contact_string**(*string*)
Convert a contact string to a matching dict.

The contact string must be in the format:

```
name <email> (url)
```

email and *url* are optional.

Example:

```
>>> r = parse_contact_string('Me <me@example.com> (http://example.com/me)')
>>> assert r == {
...     'name': 'Me',
...     'email': 'me@example.com',
...     'url': 'http://example.com/me'
... }
```

Parameters **string** (*str*) – The string to extract the contact information from.

Returns A dict which holds the extracted contact information.

Return type dict

flask_openapi.utils.**parse_werkzeug_url**(*url*)
Process a werkzeug URL rule.

Parameters **url** (*str*) – The werkzeug URL rule to process.

Returns

A tuple containing the OpenAPI formatted URL and a list of path segment descriptions.

Return type tuple

flask_openapi.utils.**ref**(**args*)

Turn a number of arguments to a JSON reference.

```
>>> ref('definitions', 'MySchema')
{'$ref': '#/definitions/MySchema'}
```

Parameters **args** (*str*) – The reference path.

Returns A JSON reference to the input path.

Return type dict

flask_openapi.validators

This module contains the OpenAPISchemaValidator.

The OpenAPISchemaValidator extends the jsonschema.Draft4Validator with the functionalities as described in Swagger schemaObject.

Contents

This package exports the public interface of Flask-OpenAPI.

f