
link.json Documentation

Release 0.11

David Delassus

September 16, 2016

1	Installation	3
2	Contents	5
2.1	API documentation	5
3	Donating	9
	Python Module Index	11

link.json provides some utilities for JSON like:

- [JSON Schema](#) (validation and generation)
- [JSON Patch](#)
- [Collection+JSON](#)
- ...

Checkout the source code on [Github](#).

Installation

```
pip install link.json
```

Contents

2.1 API documentation

2.1.1 link.json package

Submodules

link.json.collection module

class `link.json.collection.CollectionJSONResponse` (*href*, *links=None*, *items=None*, *queries=None*, *template=None*, *error=None*, **args*, ***kwargs*)

Bases: `object`

Helper class used to generate valid Collection+JSON objects.

ITEM_ID = 'id'

json ()

Generate JSON object.

Returns Collection+JSON object

Return type `dict`

classmethod **make_item** (*href*, *document*, *schema=None*)

static **template_from_schema** (*schema*)

`link.json.collection.generate_collection_response` (*href*, *links=None*, *items=None*, *queries=None*, *schema=None*, *error=None*)

Helper instantiating a `CollectionJSONResponse` class using the default schema.

Parameters

- **href** (*str*) – Base URL
- **links** (*list*) – Optional list of links
- **items** (*list*) – Optional list of items
- **queries** (*list*) – Optional list of queries
- **schema** (*dict*) – Optional item schema
- **error** (*dict*) – Optional error

Returns Collection+JSON object

Return type `dict`

link.json.exceptions module

exception `link.json.exceptions.JsonError`

Bases: `exceptions.Exception`

Base error raised in this package.

exception `link.json.exceptions.JsonTransformationError`

Bases: `link.json.exceptions.JsonError`

Transformation error.

exception `link.json.exceptions.JsonValidationError`

Bases: `link.json.exceptions.JsonError`

Validation error.

link.json.resolver module

class `link.json.resolver.JsonResolver` (*base_uri='', referrer=None, **kwargs*)

Bases: `jsonschema.validators.RefResolver`

Resolve JSON references.

See: <https://tools.ietf.org/html/draft-pbryan-zyp-json-ref-03>

resolve_remote (*uri*)

link.json.schema module

class `link.json.schema.JsonSchema` (**args, **kwargs*)

Bases: `object`

Helper class used to validate data with the JSON Schema specification.

See: <http://json-schema.org>

isvalid (*schema_or_url, data*)

Check if data is validated by schema.

Parameters

- **schema_or_url** (*dict or str*) – Schema used for validation, or URL pointing to it
- **data** (*any*) – Data to validate

Returns True if data is valid, False otherwise

Return type `bool`

validate (*schema_or_url, data*)

Validate data against schema.

Parameters

- **schema_or_url** (*dict or str*) – Schema used for validation, or URL pointing to it

- **data** (*any*) – Data to validate

Raises `JsonValidationError` – if data is not validated by schema

link.json.generator module

class `link.json.generator.SchemaGenerator`

Bases: `object`

Class used to generate JSON schema from data.

DRAFT = `'http://json-schema.org/draft-04/schema#'`

exception Error

Bases: `exceptions.Exception`

Error raised when an error occurred during schema generation.

`SchemaGenerator.get_type(obj)`

Get JSON type from object.

Parameters `obj` (*any JSON serializable basic type*) – Object to guess type name from.

Returns JSON type name

Return type `str`

Raises `SchemaGenerator.Error`: when `obj` is not a valid type

`SchemaGenerator.get_unique_keys(properties, obj, required)`

Get required keys from object.

`SchemaGenerator.process_array(obj, output=None, nested=False)`

Guess schema of array's items.

`SchemaGenerator.process_object(obj, output=None, nested=False)`

Guess schema of object's properties.

link.json.transform module

class `link.json.transform.JsonTransform(source, patch, target, *args, **kwargs)`

Bases: `object`

Apply a transformation on data.

A schema is used to validate the input data, and the output data. A JSON Patch is used to apply the transformation.

See: <http://jsonpatch.com>

patch

source

target

Module contents

Donating

I

`link.json`, [7](#)
`link.json.collection`, [5](#)
`link.json.exceptions`, [6](#)
`link.json.generator`, [7](#)
`link.json.resolver`, [6](#)
`link.json.schema`, [6](#)
`link.json.transform`, [7](#)

C

CollectionJSONResponse (class in link.json.collection), 5

D

DRAFT (link.json.generator.SchemaGenerator attribute), 7

G

generate_collection_response() (in module link.json.collection), 5

get_type() (link.json.generator.SchemaGenerator method), 7

get_unique_keys() (link.json.generator.SchemaGenerator method), 7

I

isvalid() (link.json.schema.JsonSchema method), 6

ITEM_ID (link.json.collection.CollectionJSONResponse attribute), 5

J

json() (link.json.collection.CollectionJSONResponse method), 5

JsonError, 6

JsonResolver (class in link.json.resolver), 6

JsonSchema (class in link.json.schema), 6

JsonTransform (class in link.json.transform), 7

JsonTransformationError, 6

JsonValidationError, 6

L

link.json (module), 7

link.json.collection (module), 5

link.json.exceptions (module), 6

link.json.generator (module), 7

link.json.resolver (module), 6

link.json.schema (module), 6

link.json.transform (module), 7

M

make_item() (link.json.collection.CollectionJSONResponse class method), 5

P

patch (link.json.transform.JsonTransform attribute), 7

process_array() (link.json.generator.SchemaGenerator method), 7

process_object() (link.json.generator.SchemaGenerator method), 7

R

resolve_remote() (link.json.resolver.JsonResolver method), 6

S

SchemaGenerator (class in link.json.generator), 7

SchemaGenerator.Error, 7

source (link.json.transform.JsonTransform attribute), 7

T

target (link.json.transform.JsonTransform attribute), 7

template_from_schema()
(link.json.collection.CollectionJSONResponse static method), 5

V

validate() (link.json.schema.JsonSchema method), 6