# bass Documentation

***Release 0.4.0***

## Open Permissions Platform Coalition

May 26, 2016

Contents

Bass is the identity library of the Open Permissions Platform. This is the documentation of the Python API intended for developers. To have more information on usage please refer to the documentation in our repository http://github.com/openpermissions/bass .

Bass is the identity library of the Open Permissions Platform. This is the documentation of the Python API intended for developers. To have more information on usage please refer to the documentation in our repository http://github.com/openpermissions/bass .

Bass is the identity library of the Open Permissions Platform. This is the documentation of the Python API intended for developers. To have more information on usage please refer to the documentation in our repository http://github.com/openpermissions/bass .

Bass is the identity library of the Open Permissions Platform. This is the documentation of the Python API intended for developers. To have more information on usage please refer to the documentation in our repository http://github.com/openpermissions/bass .

Contents:

CHAPTER 1

# bass package

## 1.1 Submodules

## 1.2 bass.hubkey module

Generate hub keys

The generated hub key will be of the format

<resolver_id>/<schema_version>/<hub_id>/<repository_id>/<entity_type>/<entity_id>

bass.hubkey.**generate_hub_key**(*resolver_id*, *hub_id*, *repository_id*, *entity_type*, *entity_id=None*)
Create and return an array of hub keys :param resolver_id: the service that can resolve this key :param hub_id: the unique id of the hub :param repository_id: the type of id that the provider recognises :param entity_type: the type of the entity to which the key refers. :param entity_id: ID of entity (UUID) :returns: a hub key :raises: :AttributeError: if a parameter has a bad value :TypeError: if a parameter has a bad value :ValueError: if a parameter has a bad value

bass.hubkey.**idna_encode**(*string*)
Encode a string as ASCII using IDNA so that it is a valid part of a URI

See RFC3490.

**Parameters** **string** – str

**Returns** ASCII string

bass.hubkey.**is_hub_key**(*value*)
Test if a value could be a hub key :param value: the value to test if it is a hub key :returns: True if it is a hub key

bass.hubkey.**match_part**(*string*, *part*)
Raise an exception if string doesn't match a part's regex

**Parameters**

- **string** – str
- **part** – a key in the PARTS dict

**Raises** ValueError, TypeError

bass.hubkey.**normalise_part**(*t*)

bass.hubkey.**parse_hub_key**(*key*)
Parse a hub key into a dictionary of component parts

**Parameters** **key** – str, a hub key

**Returns** dict, hub key split into parts

**Raises** ValueError

`bass.hubkey.`**`url_quote`**(*string*)
Percent encode a string as ASCII so that it is a valid part of a URI

**Parameters** **string** – str

**Returns** ASCII string

## 1.3 Module contents

CHAPTER 2

# Indices and tables

- *genindex*
- *modindex*
- *search*

## b

## B



## G



## I



## M



## N



## P



## U