

---

# **collective.restapi.pam Documentation**

***Release 1.0a***

**Mikel Larreategi**

**Apr 27, 2018**



---

## Contents

---

<b>1</b>	<b>collective.restapi.pam</b>	<b>3</b>
1.1	Documentation . . . . .	3
1.2	Installation . . . . .	3
1.3	Contribute . . . . .	3
1.4	License . . . . .	4
1.5	Contents . . . . .	4



build passing



# CHAPTER 1

---

## collective.restapi.pam

---

An add-on providing plone.restapi endpoint for translations handled using [plone.app.multilingual](#) versions 1.x and 2.x

### 1.1 Documentation

Full documentation for end users can be found in the “docs” folder, and is also available online at <https://collectiverestapipam.readthedocs.io>

### 1.2 Installation

Install collective.restapi.pam by adding it to your buildout:

```
[buildout]

...

eggs =
    collective.restapi.pam
```

and then running `bin/buildout`

### 1.3 Contribute

- Issue Tracker: <https://github.com/collective/collective.restapi.pam/issues>
- Source Code: <https://github.com/collective/collective.restapi.pam>
- Documentation: <https://collectiverestapipam.readthedocs.io>

## 1.4 License

The project is licensed under the GPLv2.

## 1.5 Contents

### 1.5.1 Multilingual Translations (p.a.multilingual)

---

**Note:** This addon will only work with `plone.app.multilingual` in Plone 4. You can install `collective.restapi.lingualplone` if you want to get similar features in Plone 4 using `Products.LinguaPlone` and use plain `plone.restapi` in Plone 5.

---

Using this addon you can get information about the translations of a content object handled using `plone.app.multilingual` versions 1.x and 2.x.

To achieve that it provides a `@translations` endpoint to handle the translation information of the content objects.

Once we have installed `plone.app.multilingual` and enabled more than one language we can link two content-items of different languages to be the translation of each other issuing a *POST* query to the `@translations` endpoint including the *id* of the content which should be linked to. The *id* of the content must be a full URL of the content object: `http`

```
POST /plone/en/test-document/@translations HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46c2VjcmV0
Content-Type: application/json

{
  "id": "http://localhost:55001/plone/es/test-document"
}
```

`curl`

```
curl -i -X POST http://nohost/plone/en/test-document/@translations -H "Accept: application/json" -H "Content-Type: application/json" --data-raw '{"id": "http://localhost:55001/plone/es/test-document"}' --user admin:secret
```

`httpie`

```
http -j POST http://nohost/plone/en/test-document/@translations id=http://localhost:55001/plone/es/test-document -a admin:secret
```

`python-requests`

```
requests.post('http://nohost/plone/en/test-document/@translations', headers={'Accept': 'application/json'}, json={'id': 'http://localhost:55001/plone/es/test-document'}, auth=('admin', 'secret'))
```

---

**Note:** “id” is a required field and needs to point to an existing content on the site.

---

The API will return a *201 Created* response if the linking was successful.



```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://localhost:55001/plone/en/test-document

{}
```

After linking the contents we can get the list of the translations of that content item by issuing a GET request on the `@translations` endpoint of that content item.: http

```
GET /plone/en/test-document/@translations HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46c2VjcmV0
```

curl

```
curl -i http://nohost/plone/en/test-document/@translations -H "Accept: application/
↪ json" --user admin:secret
```

httpie

```
http -j http://nohost/plone/en/test-document/@translations -a admin:secret
```

python-requests

```
requests.get('http://nohost/plone/en/test-document/@translations', headers={'Accept':
↪ 'application/json'}, auth=('admin', 'secret'))
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "@id": "http://localhost:55001/plone/en/test-document/@translations",
  "items": [
    {
      "@id": "http://localhost:55001/plone/es/test-document",
      "language": "es"
    }
  ]
}
```

To unlink the content, issue a DELETE request on the `@translations` endpoint of the content item and provide the language code you want to unlink.: http

```
DELETE /plone/en/test-document/@translations HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46c2VjcmV0
Content-Type: application/json

{
  "language": "es"
}
```

curl

```
curl -i -X DELETE http://nohost/plone/en/test-document/@translations -H "Accept:
↪ application/json" -H "Content-Type: application/json" --data-raw '{"language": "es"}'
↪ --user admin:secret
```

httpie

```
http -j DELETE http://nohost/plone/en/test-document/@translations language=es -a_
↪admin:secret
```

python-requests

```
requests.delete('http://nohost/plone/en/test-document/@translations', headers={'Accept': 'application/json'}, json={'language': 'es'}, auth=('admin', 'secret'))
```

---

**Note:** “language” is a required field.

---

```
HTTP/1.1 204 No Content
```

## Expansion

This endpoint uses the `expansion` mechanism of `plone.restapi` which allows to get additional information about a content item in one query, avoiding unnecessary requests.

If a simple GET request is done on the content item, a new entry will be shown on the `@components` entry with the URL of the `@translations` endpoint: `http`

```
GET /plone/en/test-document HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46c2VjcmV0
```

curl

```
curl -i http://nohost/plone/en/test-document -H "Accept: application/json" --user_
↪admin:secret
```

httpie

```
http -j http://nohost/plone/en/test-document -a admin:secret
```

python-requests

```
requests.get('http://nohost/plone/en/test-document', headers={'Accept': 'application/json'}, auth=('admin', 'secret'))
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "@components": {
    "breadcrumbs": {
      "@id": "http://localhost:55001/plone/en/test-document/@breadcrumbs"
    },
    "navigation": {
      "@id": "http://localhost:55001/plone/en/test-document/@navigation"
    },
    "translations": {
      "@id": "http://localhost:55001/plone/en/test-document/@translations"
    },
  },
}
```

```

    "workflow": {
      "@id": "http://localhost:55001/plone/en/test-document/@workflow"
    },
    "@id": "http://localhost:55001/plone/en/test-document",
    "@type": "Document",
    "UID": "SomeUUID00000000000000000000000000000009",
    "allow_discussion": null,
    "changeNote": "",
    "contributors": [],
    "created": "2016-10-21T21:00:00+02:00",
    "creators": [
      "test_user_1_",
      "admin"
    ],
    "description": "",
    "effective": null,
    "exclude_from_nav": false,
    "expires": null,
    "id": "test-document",
    "is_folderish": false,
    "language": "en",
    "layout": "document_view",
    "modified": "2016-10-21T21:00:00+02:00",
    "parent": {
      "@id": "http://localhost:55001/plone/en",
      "@type": "LRF",
      "description": "",
      "review_state": "published",
      "title": "English"
    },
    "relatedItems": [],
    "review_state": "private",
    "rights": "",
    "subjects": [],
    "table_of_contents": null,
    "text": null,
    "title": "Test document",
    "version": "current"
  }
}

```

That means that, to include the translations of the item in the main content response, you can request to expand it: http

```

GET /plone/en/test-document?expand=translations HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46c2VjcmV0

```

curl

```

curl -i 'http://nohost/plone/en/test-document?expand=translations' -H "Accept:
↪application/json" --user admin:secret

```

httpie

```

http -j 'http://nohost/plone/en/test-document?expand=translations' -a admin:secret

```

python-requests

```
requests.get('http://nohost/plone/en/test-document?expand=translations', headers={
    ↪ 'Accept': 'application/json'}, auth=('admin', 'secret'))
```

And the response will include the required information:

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "@components": {
    "breadcrumbs": {
      "@id": "http://localhost:55001/plone/en/test-document/@breadcrumbs"
    },
    "navigation": {
      "@id": "http://localhost:55001/plone/en/test-document/@navigation"
    },
    "translations": {
      "@id": "http://localhost:55001/plone/en/test-document/@translations",
      "items": [
        {
          "@id": "http://localhost:55001/plone/es/test-document",
          "language": "es"
        }
      ]
    },
    "workflow": {
      "@id": "http://localhost:55001/plone/en/test-document/@workflow"
    }
  },
  "@id": "http://localhost:55001/plone/en/test-document",
  "@type": "Document",
  "UID": "SomeUUID000000000000000000000009",
  "allow_discussion": null,
  "changeNote": "",
  "contributors": [],
  "created": "2016-10-21T21:00:00+02:00",
  "creators": [
    "test_user_1_",
    "admin"
  ],
  "description": "",
  "effective": null,
  "exclude_from_nav": false,
  "expires": null,
  "id": "test-document",
  "is_folderish": false,
  "language": "en",
  "layout": "document_view",
  "modified": "2016-10-21T21:00:00+02:00",
  "parent": {
    "@id": "http://localhost:55001/plone/en",
    "@type": "LRF",
    "description": "",
    "review_state": "published",
    "title": "English"
  },
  "relatedItems": [],
  "review_state": "private",
```

```

"rights": "",
"subjects": [],
"table_of_contents": null,
"text": null,
"title": "Test document",
"version": "current"
}

```

## 1.5.2 Upgrade guide

This upgrade guide lists all breaking changes in collective.restapi.pam and explains the necessary steps that are needed to upgrade to the latest version.

### Upgrading to collective.restapi.pam 2.0.0

When using the `@translations` endpoint in collective.restapi.pam 1.x, the endpoint returned a *language* key with the content object's language and a *translations* key with all its translations.

Now, as the endpoint is expandable we want the endpoint to behave like the other expandable endpoints. As top level information we only include the name of the endpoint on the `@id` attribute and the actual translations of the content object in an attribute called *items*.

This means that now the JSON response to a GET request to the *Multilingual Translations (p.a.multilingual)* endpoint does not include anymore the language of the actual content item and the translations in an attribute called *items* instead of *translations*.

Old response:

```

HTTP/1.1 200 OK
Content-Type: application/json

{
  "@id": "http://localhost:55001/plone/en/test-document",
  "language": "en",
  "translations": [
    {
      "@id": "http://localhost:55001/plone/es/test-document",
      "language": "es"
    }
  ]
}

```

New response:

```

HTTP/1.1 200 OK
Content-Type: application/json

{
  "@id": "http://localhost:55001/plone/en/test-document/@translations",
  "items": [
    {
      "@id": "http://localhost:55001/plone/es/test-document",
      "language": "es"
    }
  ]
}

```