
python-almanachclient

Release

June 30, 2017

1	Almanach API Client	3
2	Installation	5
2.1	Requirements	5
2.2	Instructions	5
3	Command Line Usage	7
3.1	Environment variables	7
3.2	Optional Arguments	7
3.3	Get server version	7
3.4	Get Endpoint URL	8
3.5	Get Tenant Entities	8
3.6	Get one Entity	8
3.7	List Instances Entities	8
3.8	Create Instance Entity	9
3.9	Update Instance Entity	9
3.10	Delete Instance	10
3.11	Resize Instance	10
3.12	List Volumes	10
3.13	Create Volume	11
3.14	Delete Volume	11
3.15	Resize Volume	11
3.16	Attach Volume	12
3.17	Detach Volume	12
3.18	List Volume Types	12
3.19	Get Volume Type	13
3.20	Create Volume Type	13
3.21	Delete Volume Type	13
4	Almanach Python API	15
4.1	Usage	15
4.2	Examples	15
4.3	Reference	16
5	Contributing	21
	Python Module Index	23

Contents:

Almanach API Client

This is a client library for Almanach built on the Almanach REST API.

It provides a Python API (The `almanachclient` module) and a command-line tool (`almanach-client`).

- Free software: Apache license
- Source: <http://git.openstack.org/cgit/openstack/python-almanachclient>
- Bugs: <http://bugs.launchpad.net/almanach>
- Documentation: <https://python-almanachclient.readthedocs.io/>

Installation

Requirements

- Python $\geq$ 3.4
- Almanach $\geq$ 4.0.9

Instructions

We strongly encourage to use a virtual environment to install the software: `python3 -m venv my_virtual_env`.

At the command line:

```
$ pip install python-almanachclient
```

Command Line Usage

Environment variables

- `OS_AUTH_URL`: Keystone URL (v3 endpoint)
- `OS_REGION_NAME`: OpenStack region name
- `OS_USERNAME`: OpenStack username
- `OS_PASSWORD`: OpenStack password
- `ALMANACH_SERVICE`: Almanach Keystone catalog service name
- `ALMANACH_TOKEN`: Almanach API token, if empty a token will be fetched from Keystone
- `ALMANACH_URL`: Almanach API base URL, if empty the endpoint will be fetched from Keystone catalog

Optional Arguments

- `--os-auth-url`: Keystone URL (v3 endpoint)
- `--os-region-name`: OpenStack region name
- `--os-username`: OpenStack username
- `--os-password`: OpenStack password
- `--almanach-service`: Almanach Keystone catalog service name
- `--almanach-token`: Almanach API token, if empty a token will be fetched from Keystone
- `--almanach-url`: Almanach API base URL, if empty the endpoint will be fetched from Keystone catalog

Get server version

Usage: `almanach version`

`almanach version`

4.0.9

Get Endpoint URL

Usage: almanach endpoint

almanach endpoint

http://almanach.example.org

Get Tenant Entities

Usage: almanach list-entities <tenant_id> <start> --end <end>

almanach list-entities bca89ae64dba46b8b74653d8d9ae8364 2016-01-01 --end 2017-05-30

Entity ID	Type	Name	Start	End	Properties
8c3bc3aa-28d6-4863-b5ae-72e1b415f79d	instance	vm01	2017-05-09 14:19:14+00:00	None	{'image': 'ubuntu-14.04', 'size': 1}
f0690323-c394-4848-a272-964aad6431aa	instance	vm02	2017-05-15 18:31:42+00:00	None	{'image': 'ubuntu-14.04', 'size': 1}
3e3b22e6-a10c-4c00-b8e5-05fcc8422b11	volume	vol01	2017-05-15 19:11:14+00:00	None	{'size': 1, 'type': 'volume'}

Arguments:

- tenant_id: Tenant ID (UUID)
- start: Start date (ISO8601 format)
- end: End date (ISO8601 format), optional

Get one Entity

Usage: almanach get-entity <entity_id>

almanach get-entity 3e3b22e6-a10c-4c00-b8e5-05fcc8422b11

Tenant ID	Type	Name	Start	End	Properties
bca89ae64dba46b8b74653d8d9ae8364	volume	vol1	2017-05-15 19:11:14+00:00	None	{'size': 1, 'type': 'volume'}

Arguments:

- entity_id: Entity ID (UUID)

List Instances Entities

Usage: almanach list-instances <tenant_id> <start> <end>

almanach list-instances bca89ae64dba46b8b74653d8d9ae8364 2016-01-01 2017-05-30

Instance ID	Name	Start	End
-------------	------	-------	-----

f0690323-c394-4848-a272-964aad6431aa	vm02	2017-05-15 18:31:42+00:00	None
8c3bc3aa-28d6-4863-b5ae-72e1b415f79d	vm01	2017-05-09 14:19:14+00:00	2017-05-17 09:37:47.779

Arguments:

- tenant_id: Tenant ID (UUID)
- start: Start date (ISO8601 format)
- end: End date (ISO8601 format)

Create Instance Entity

Usage: `almanach create_instance <tenant_id> <instance_id> <name> <flavor> <start> --image-meta <image_meta>`

Example:

```
almanach create-instance bca89ae64dba46b8b74653d8d9ae8364 \
    8d8d0dc7-5f06-40aa-aba8-c4ff02aeb866 \
    my-instance \
    my-flavor \
    2017-01-01 \
    --image-meta '{"distro": "centos7", "type": "linux"}'
```

Success

- tenant_id: Tenant ID (UUID)
- instance_id: Instance ID (UUID)
- start: Start date (ISO8601 format)
- name: Instance name (string)
- flavor: Flavor (string)
- image_meta: Image metadata (dict as JSON string)

Update Instance Entity

Usage: `almanach update-instance <instance_id> --start <start> --end <end> --name <name> --flavor <flavor>`

```
almanach update-instance 8c3bc3aa-28d6-4863-b5ae-72e1b415f79d --name vm03
```

Field	Value
Tenant ID	bca89ae64dba46b8b74653d8d9ae8364
Instance ID	8c3bc3aa-28d6-4863-b5ae-72e1b415f79d
Start	2017-05-09 14:19:14+00:00
End	None
Name	vm03
Flavor	A1.1

```
| Image          | {'distro': 'centos', 'version': '7', 'os_type': 'linux'} |
+-----+-----+
```

Arguments:

- instance_id: Instance ID (UUID)
- start: Start date (ISO8601 format)
- end: End date (ISO8601 format)
- name: Instance name (string)
- flavor: Flavor (string)

Delete Instance

Usage: almanach delete-instance <instance_id> --end <end>

```
almanach delete-instance 8c3bc3aa-28d6-4863-b5ae-72e1b415f79d
```

Success

Arguments:

- instance_id: Instance ID (UUID)
- end: End date (ISO8601 format), if not specified the current date time is used

Resize Instance

Usage: almanach resize-instance <instance_id> <flavor> --date <resize_date>

```
almanach resize-instance 8c3bc3aa-28d6-4863-b5ae-72e1b415f79d New_Flavor
```

Success

Arguments:

- instance_id: Instance ID (UUID)
- flavor: Flavor (string)
- date: Resize date (ISO8601 format), if not specified the current datetime is used

List Volumes

Usage: almanach list-volumes <tenant_id> <start> <end>

```
almanach list-volumes bca89ae64dba46b8b74653d8d9ae8364 2016-01-01 2017-09-01
```

```
+-----+-----+-----+-----+-----+-----+
| Volume ID          | Name | Start                          | End | Type          | Size |
+-----+-----+-----+-----+-----+-----+
| 3e3b22e6-a10c-4c00-b8e5-05fcc8422b11 | vol1 | 2017-05-15 19:11:14+00:00 | None | solidfire0 | 1    |
+-----+-----+-----+-----+-----+-----+
```

Arguments:

- `tenant_id`: Tenant ID (UUID)
- `start`: Start date (ISO8601 format)
- `end`: End date (ISO8601 format)

Create Volume

Usage: `almanach create-volume <tenant_id> <volume_id> <volume_type_id> <volume_name> <size> --date <creation_date> --attachment <instance_id>`

```
almanach create-volume \
  8c3bc3aa-28d6-4863-b5ae-72e1b415f79d \
  3e3b22e6-a10c-4c00-b8e5-05fcc8422b11 \
  f3786e9f-f8e6-4944-a3bc-e11b9f112706 \
  my-volume \
  5 \
  --attachment=86dd5189-d9d6-40f7-a319-19231fbd4e07 \
  --attachment=252e49d8-abf2-486c-8478-b5f775134f54
```

Success

Arguments:

- `tenant_id`: Tenant ID (UUID)
- `volume_id`: Volume ID (UUID)
- `volume_type_id`: Volume ID (UUID)
- `volume_name`: Volume name (string)
- `size`: Volume size (integer)
- `date`: Creation date (ISO8601 format), if not specified the current datetime is used
- `attachment`: Attach the volume to one or many instances (UUID)

Delete Volume

Usage: `almanach delete-volume <volume_id> --end <end>`

```
almanach delete-volume 8c3bc3aa-28d6-4863-b5ae-72e1b415f79d
```

Success

Arguments:

- `volume_id`: Instance ID (UUID)
- `end`: End date (ISO8601 format), if not specified the current date time is used

Resize Volume

Usage: `almanach resize-volume <volume_id> <size> --date <resize_date>`

```
almanach resize-volume 8c3bc3aa-28d6-4863-b5ae-72e1b415f79d 2
```

Success

Arguments:

- `volume_id`: Volume ID (UUID)
- `size`: Volume size (integer)
- `date`: Resize date (ISO8601 format), if not specified the current datetime is used

Attach Volume

Usage: `almanach attach-volume <volume_id> --date <creation_date> --attachment <instance_id>`

```
almanach attach-volume \  
    8c3bc3aa-28d6-4863-b5ae-72e1b415f79d \  
    --attachment=86dd5189-d9d6-40f7-a319-19231fbd4e07
```

Success

Arguments:

- `volume_id`: Volume ID (UUID)
- `date`: Attachment date (ISO8601 format), if not specified the current datetime is used
- `attachment`: Attach the volume to one or many instances (UUID)

Detach Volume

Usage: `almanach detach-volume <volume_id> --date <creation_date> --attachment <instance_id>`

```
almanach detach-volume \  
    8c3bc3aa-28d6-4863-b5ae-72e1b415f79d \  
    --attachment=86dd5189-d9d6-40f7-a319-19231fbd4e07
```

Success

Arguments:

- `volume_id`: Volume ID (UUID)
- `date`: Attachment date (ISO8601 format), if not specified the current datetime is used
- `attachment`: Attach the volume to one or many instances (UUID)

List Volume Types

Usage: `almanach list-volume-types`

```
almanach list-volume-types
```

```
+-----+-----+
| Volume Type ID           | Volume Type Name |
+-----+-----+
| f3786e9f-f8e6-4944-a3bc-e11b9f112706 | solidfire0       |
+-----+-----+
```

Get Volume Type

Usage: almanach get-volume-type <volume_type_id>

```
almanach get-volume-type f3786e9f-f8e6-4944-a3bc-e11b9f112706
```

```
+-----+-----+
| Field           | Value                                     |
+-----+-----+
| Volume Type ID  | f3786e9f-f8e6-4944-a3bc-e11b9f112706 |
| Volume Type Name | solidfire0                             |
+-----+-----+
```

Create Volume Type

Usage: almanach create-volume-type <volume_type_id> <volume_type_name>

```
almanach create-volume-type f1c2db7b-946e-47a4-b443-914a669a6672 my_volume_type
```

```
Success
```

Delete Volume Type

Usage: almanach delete-volume-type <volume_type_id>

```
almanach delete-volume-type f1c2db7b-946e-47a4-b443-914a669a6672
```

```
Success
```

Almanach Python API

Usage

First, create an Almanach Client instance by providing a Keystone session and your AUTH_TOKEN:

```
>>> from keystoneauth1 import loading
>>> from keystoneauth1 import session
>>> from almanachclient.v1.client import Client

>>> loader = loading.get_plugin_loader('password')
>>> auth = loader.load_from_options(auth_url=AUTH_URL,
...                               username=USERNAME,
...                               password=PASSWORD,
...                               project_name=PROJECT_NAME,
...                               project_domain_name="Default",
...                               user_domain_name="Default")
>>> sess = session.Session(auth=auth)

>>> almanach = Client(session=sess, token=AUTH_TOKEN)
```

Here ALMANACH_URL will be a string that represents the url of Almanach API. AUTH_TOKEN will be the authorization token you use to access the API.

You can also create a Keystone Client instance to fetch Almanach API endpoint:

```
>>> from almanachclient.keystone_client import KeystoneClient
>>> keystone_client = KeystoneClient(KEYSTONE_URL, USERNAME, PASSWORD, SERVICE, REGION)
>>> almanach_url = keystone_client.get_endpoint_url()
```

In this case KEYSTONE_URL will be a string that represents the url of your keystone catalog. The nature of USERNAME and PASSWORD speak for themselves. SERVICE will be a string you use to identify the “almanach” service. REGION is used to determine which region the “almanach” service is being used in.

Examples

```
>>> import datetime
>>> from almanachclient.v1.client import Client

>>> start_date = datetime.datetime(2017, 01, 05)
>>> end_date = datetime.datetime(2017, 02, 05)
```

```
>>> almanach = Client('http://api.region.example.org', 'myApiAuthorizationToken')
>>> almanach.get_info()

>>> almanach.get_volume_types()
>>> almanach.get_volume_type('f1c2db7b-946e-47a4-b443-914a669a6673')
>>> almanach.create_volume_type('f1c2db7b-946e-47a4-b443-914a669a5555', 'VolumeTypeName')
>>> almanach.delete_volume_type('f1c2db7b-946e-47a4-b443-914a669a5555')

>>> almanach.get_tenant_entities('my-tenant-uuid', start_date, end_date)

>>> almanach.delete_instance('f1c2db7b-946e-47a4-b443-914a669a3333')
>>> almanach.update_instance_entity(instance_id='f1c2db7b-946e-47a4-b443-914a669a2222', start=start_date, end=end_date)
```

Reference

For more information, see the reference:

The `almanachclient.exceptions` Module

exception `almanachclient.exceptions.ClientError`

Bases: `builtins.Exception`

exception `almanachclient.exceptions.EndpointNotFound`

Bases: `almanachclient.exceptions.ClientError`

exception `almanachclient.exceptions.HTTPError`

Bases: `almanachclient.exceptions.ClientError`

The `almanachclient.keystone_client` Module

class `almanachclient.keystone_client.KeystoneClient` (*auth_url*, *username*, *password*, *service*, *region_name*, *domain_name*=*'default'*, *user_domain_id*=*'default'*)

Bases: `builtins.object`

get_endpoint_url (*visibility*=*'admin'*)

Get Almanach API URL from Keystone catalog

Parameters *visibility* (*str*) – Service visibility

Returns Almanach Endpoint URL

Return type *str*

get_token ()

Get Keystone token

Return type *str*

The almanachclient.v1.client Module

```
class almanachclient.v1.client.Client (url=None, token=None, session=None, re-
                                     gion_name=None, service_type='cloudmetrics', end-
                                     point_type=None)
```

Bases: almanachclient.http_client.HttpClient

DATE_FORMAT_BODY = '%Y-%m-%dT%H:%M:%S.%fZ'

DATE_FORMAT_QS = '%Y-%m-%d %H:%M:%S.%f'

api_version = 'v1'

attach_volume (volume_id, attachments, attachment_date=None)

Attach instances to a volume.

Parameters

- **volume_id** (*str*) – Volume UUID
- **attachments** (*list*) – List of instance ID
- **attachment_date** (*datetime*) – Attachment date

Raises ClientError

Return type bool

create_instance (tenant_id, instance_id, name, flavor, start, image_meta=None)

Create an instance for a tenant.

Parameters

- **tenant_id** (*str*) – The Tenant UUID
- **instance_id** (*str*) – The instance UUID
- **name** (*str*) – The instance name
- **flavor** (*str*) – The flavor
- **start** (*datetime*) – Start date
- **image_meta** (*dict*) – The OS type, distro and version of the image

Raises ClientError

Return type bool

create_volume (tenant_id, volume_id, volume_type_id, name, size, attachments=None, start=None)

Create a volume.

Parameters

- **tenant_id** (*str*) – Tenant UUID
- **volume_id** (*str*) – Volume UUID
- **volume_type_id** (*str*) – Volume type
- **name** (*str*) – Volume name
- **size** (*int*) – Volume size
- **attachments** (*list*) – List of instance attached to the volume
- **start** (*datetime*) – Creation date or now if None

Raises ClientError

Return type bool

create_volume_type (*volume_type_id*, *volume_type_name*)
Create a volume type.

Parameters

- **volume_type_id** (*str*) – The Volume Type ID
- **volume_type_name** (*str*) – The Volume Type name

Raises ClientError

Return type bool

delete_instance (*instance_id*, *end=None*)
Delete an instance.

Parameters

- **instance_id** (*str*) – Instance UUID
- **end** (*datetime*) – End date

Raises ClientError

Return type bool

delete_volume (*volume_id*, *end=None*)
Remove a volume.

Parameters

- **volume_id** (*str*) – Volume UUID
- **end** (*datetime*) – Suppression date

Raises ClientError

Return type bool

delete_volume_type (*volume_type_id*)
Delete the volume type.

Parameters **volume_type_id** (*str*) – Volume Type UUID

Raises ClientError

Return type bool

detach_volume (*volume_id*, *attachments*, *attachment_date=None*)
Detach instances from a volume.

Parameters

- **volume_id** (*str*) – Volume UUID
- **attachments** (*list*) – List of instance ID
- **attachment_date** (*datetime*) – Attachment date

Raises ClientError

Return type bool

get_entity (*entity_id*)
Get single entity.

Parameters **entity_id** (*str*) – Entity UUID

Raises ClientError

Return type list

get_info()

Display information about the current version and counts of entities in the database.

Return type dict

get_instances (*tenant_id*, *start*, *end*)

List instances for a tenant.

Parameters

- **tenant_id** (*str*) – The Tenant UUID
- **start** (*datetime*) – Start date
- **end** (*datetime*) – End date

Raises ClientError

Return type list

get_tenant_entities (*tenant_id*, *start*, *end=None*)

List instances and volumes for a tenant.

Parameters

- **tenant_id** (*str*) – Tenant UUID
- **start** (*datetime*) – Start date
- **end** (*datetime*) – End date

Raises ClientError

Return type list

get_url()

get_volume_type (*volume_type_id*)

Get a volume type.

Parameters **volume_type_id** (*str*) – Volume Type UUID

Raises ClientError

Return type dict

get_volume_types()

List volume types.

Return type list

get_volumes (*tenant_id*, *start*, *end*)

List volumes for a tenant.

Parameters

- **tenant_id** (*str*) – The Tenant UUID
- **start** (*datetime*) – Start date
- **end** (*datetime*) – End date

Raises ClientError

Return type list

resize_instance (*instance_id*, *flavor*, *resize_date=None*)

Resize an instance.

Parameters

- **instance_id** (*str*) – Instance UUID
- **flavor** (*str*) – Flavor
- **resize_date** (*datetime*) – Resize date

Raises ClientError

Return type bool

resize_volume (*volume_id*, *size*, *resize_date=None*)

Resize a volume.

Parameters

- **volume_id** (*str*) – Volume UUID
- **size** (*int*) – Volume size
- **resize_date** (*datetime*) – Resize date

Raises ClientError

Return type bool

update_instance_entity (*instance_id*, ***kwargs*)

Update an instance entity.

Parameters

- **instance_id** (*str*) – Instance UUID
- **start_date** (*datetime*) – Start date
- **end_date** (*datetime*) – End date
- **flavor** (*str*) – The flavor
- **name** (*str*) – The instance name

Raises ClientError

Return type dict

Contributing

If you would like to contribute to the development of OpenStack, you must follow the steps in this page:

<http://docs.openstack.org/infra/manual/developers.html>

If you already have a good understanding of how the system works and your OpenStack accounts are set up, you can skip to the development workflow section of this documentation to learn how changes to OpenStack should be submitted for review via the Gerrit tool:

<http://docs.openstack.org/infra/manual/developers.html#development-workflow>

Pull requests submitted through GitHub will be ignored.

Bugs should be filed on Launchpad, not GitHub:

<https://bugs.launchpad.net/almanach>

a

`almanachclient`, [15](#)
`almanachclient.exceptions`, [16](#)
`almanachclient.keystone_client`, [16](#)
`almanachclient.v1.client`, [17](#)

A

almanachclient (module), 15
almanachclient.exceptions (module), 16
almanachclient.keystone_client (module), 16
almanachclient.v1.client (module), 17
api_version (almanachclient.v1.client.Client attribute), 17
attach_volume() (almanachclient.v1.client.Client method), 17

C

Client (class in almanachclient.v1.client), 17
ClientError, 16
create_instance() (almanachclient.v1.client.Client method), 17
create_volume() (almanachclient.v1.client.Client method), 17
create_volume_type() (almanachclient.v1.client.Client method), 18

D

DATE_FORMAT_BODY (almanachclient.v1.client.Client attribute), 17
DATE_FORMAT_QS (almanachclient.v1.client.Client attribute), 17
delete_instance() (almanachclient.v1.client.Client method), 18
delete_volume() (almanachclient.v1.client.Client method), 18
delete_volume_type() (almanachclient.v1.client.Client method), 18
detach_volume() (almanachclient.v1.client.Client method), 18

E

EndpointNotFound, 16

G

get_endpoint_url() (almanachclient.keystone_client.KeystoneClient method), 16

get_entity() (almanachclient.v1.client.Client method), 18
get_info() (almanachclient.v1.client.Client method), 19
get_instances() (almanachclient.v1.client.Client method), 19
get_tenant_entities() (almanachclient.v1.client.Client method), 19
get_token() (almanachclient.keystone_client.KeystoneClient method), 16
get_url() (almanachclient.v1.client.Client method), 19
get_volume_type() (almanachclient.v1.client.Client method), 19
get_volume_types() (almanachclient.v1.client.Client method), 19
get_volumes() (almanachclient.v1.client.Client method), 19

H

HTTPError, 16

K

KeystoneClient (class in almanachclient.keystone_client), 16

R

resize_instance() (almanachclient.v1.client.Client method), 19
resize_volume() (almanachclient.v1.client.Client method), 20

U

update_instance_entity() (almanachclient.v1.client.Client method), 20