Octavia Dashboard Documentation

Release 15.0.1.dev6

OpenStack Octavia Team

Mar 25, 2026

CONTENTS

CHAPTER

ONE

OCTAVIA DASHBOARD

1.1 Team and repository tags

1.2 octavia-dashboard

Horizon panels for Octavia

- Free software: Apache license
- Documentation: https://docs.openstack.org/octavia-dashboard/latest/
- Source: https://opendev.org/openstack/octavia-dashboard
- Release notes: https://docs.openstack.org/releasenotes/octavia-dashboard/
- Bugs: https://storyboard.openstack.org/#!/project/909

1.2.1 Features

- Please see octavia repository

1.2.2 Howto

1. Package the octavia_dashboard by running:

   ```
   python setup.py sdist
   ```

   This will create a python egg in the dist folder, which can be used to install on the horizon machine or within horizon's python virtual environment.

2. Copy `_1482_project_load_balancer_panel.py` in `octavia_dashboard/enabled` directory to `openstack_dashboard/local/enabled`:

   ```
   $ cp -a \
     ${OCTAVIA_DASHBOARD_DIR}/octavia_dashboard/enabled/_1482_*.py \
     ${HORIZON_DIR}/openstack_dashboard/local/enabled/
   ```

3. (Optional) Generate the policy file and copy into horizon's policy files folder, and copy `_1499_load_balancer_settings.py` in `octavia_dashboard/local_settings.d` directory to `openstack_dashboard/local/local_settings.d`:

```
$ oslopolicy-policy-generator \
  --config-file \
  ${OCTAVIA_DIR}/etc/policy/octavia-policy-generator.conf \
  --output-file \
  ${OCTAVIA_DASHBOARD_DIR}/octavia_dashboard/conf/octavia_policy.yaml
$ cp -a \
  ${OCTAVIA_DASHBOARD_DIR}/octavia_dashboard/conf/octavia_policy.yaml \
  ${HORIZON_DIR}/openstack_dashboard/conf/
$ cp -a \
  ${OCTAVIA_DASHBOARD_DIR}/octavia_dashboard/local_settings.d/_1499_*.py \
  ${HORIZON_DIR}/openstack_dashboard/local/local_settings.d/
```

4. Django has a compressor feature that performs many enhancements for the delivery of static files. If the compressor feature is enabled in your environment (`COMPRESS_OFFLINE = True`), run the following commands:

```
$ ./manage.py collectstatic
$ ./manage.py compress
```

5. Finally restart your web server to enable octavia-dashboard in your Horizon:

```
$ sudo service apache2 restart
```

CHAPTER

TWO

INSTALLATION

At the command line:

```
$ pip install octavia-dashboard
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv octavia-dashboard
$ pip install octavia-dashboard
```

To enable the panels in Horizon, copy _1482_project_load_balancer_panel.py in octavia_dashboard/enabled directory to openstack_dashboard/local/enabled

(Optional) To enable policy enforcement at the Horizon level, copy the policy file into horizon's policy files folder, and add this config `POLICY_FILES`:

```
'octavia': 'octavia_policy.json',
```

Django has a compressor feature that performs many enhancements for the delivery of static files. If the compressor feature is enabled in your environment (`COMPRESS_OFFLINE = True`), run the following commands:

```
$ ./manage.py collectstatic
$ ./manage.py compress
```

Finally restart your web server to enable octavia-dashboard in your Horizon:

Ubuntu:

```
$ sudo service apache2 restart
```

Red Hat based:

```
$ sudo systemctl restart httpd
```

CHAPTER

THREE

CONTRIBUTING

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

> https://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:

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

Pull requests submitted through GitHub will be ignored.

Bugs should be filed on Storyboard, not GitHub or Launchpad:

> https://storyboard.openstack.org/#!/project/909

CHAPTER

FOUR

REFERENCE

4.1 Module Reference

4.1.1 octavia_dashboard

octavia_dashboard package

Subpackages

octavia_dashboard.api package

Subpackages

octavia_dashboard.api.rest package

Submodules

octavia_dashboard.api.rest.barbican module

API over the barbican service.

class octavia_dashboard.api.rest.barbican.**SSLCertificates**(***kwargs*)

Bases: View

API for working with SSL certificate containers.

get(*request*)

List certificate containers.

The listing result is an object with property "items".

url_regex = 'octavia-barbican/certificates/$'

class octavia_dashboard.api.rest.barbican.**Secrets**(***kwargs*)

Bases: View

API for working with secrets.

get(*request*)

List secrets.

The listing result is an object with property "items".

url_regex = 'octavia-barbican/secrets/$'

octavia_dashboard.api.rest.lbaasv2 module

API over the neutron LBaaS v2 service.

class octavia_dashboard.api.rest.lbaasv2.**AvailabilityZones**(***kwargs*)

Bases: View

API for load balancer availability zones.

get(*request*)

List of availability zones for the current project.

The listing result is an object with property "items".

url_regex = 'lbaas/availabilityzones/$'

class octavia_dashboard.api.rest.lbaasv2.**Flavor**(***kwargs*)

Bases: View

API for retrieving a single flavor.

delete(*request*, *flavor_id*)

Delete a specific flavor.

http://localhost/api/lbaas/flavors/3971d368-ca9b-4770-929a-3adca5bf89eb

get(*request*, *flavor_id*)

Get a specific flavor.

put(*request*, *flavor_id*)

Edit a flavor.

url_regex = 'lbaas/flavors/(?P<flavor_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**FlavorProfile**(***kwargs*)

Bases: View

API for retrieving a single flavor profile.

delete(*request*, *flavor_profile_id*)

Delete a specific flavor profile.

http://localhost/api/lbaas/flavorprofiles/e8150eab-aefa-42cc-867e-3fb336da52bd

get(*request*, *flavor_profile_id*)

Get a specific flavor profile.

put(*request*, *flavor_profile_id*)

Edit a flavor profile.

url_regex = 'lbaas/flavorprofiles/(?P<flavor_profile_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**FlavorProfiles**(***kwargs*)

Bases: View

API for load balancer flavor profiles.

get(*request*)

List of flavor profiles for the current project.

The listing result is an object with property "items".

post(*request*)

Create a new flavor_profile.

url_regex = 'lbaas/flavorprofiles/$'

class octavia_dashboard.api.rest.lbaasv2.**Flavors**(***kwargs*)

Bases: View

API for load balancer flavors.

get(*request*)

List of flavors for the current project.

The listing result is an object with property "items".

post(*request*)

Create a new flavor.

url_regex = 'lbaas/flavors/$'

class octavia_dashboard.api.rest.lbaasv2.**HealthMonitor**(***kwargs*)

Bases: View

API for retrieving a single health monitor.

delete(*request*, *health_monitor_id*)

Delete a specific health monitor.

http://localhost/api/lbaas/healthmonitors/cc758c90-3d98-4ea1-af44-aab405c9c915

get(*request*, *health_monitor_id*)

Get a specific health monitor.

put(*request*, *health_monitor_id*)

Edit a health monitor.

url_regex = 'lbaas/healthmonitors/(?P<health_monitor_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**HealthMonitors**(***kwargs*)

Bases: View

API for load balancer pool health monitors.

get(*request*)

List of health monitors for the current project.

The listing result is an object with property "items".

post(*request*)

Create a new health monitor.

url_regex = 'lbaas/healthmonitors/$'

class octavia_dashboard.api.rest.lbaasv2.**L7Policies**(***kwargs*)

Bases: View

API for load balancer l7 policies.

get(*request*)

List of l7 policies for the current project.

The listing result is an object with property "items".

post(*request*)

Create a new l7 policy.

Creates a new l7 policy as well as other optional resources such as l7 rules.

url_regex = 'lbaas/l7policies/$'

class octavia_dashboard.api.rest.lbaasv2.**L7Policy**(***kwargs*)

Bases: View

API for retrieving a single l7 policy.

delete(*request*, *l7_policy_id*)

Delete a specific l7 policy.

http://localhost/api/lbaas/l7policies/cc758c90-3d98-4ea1-af44-aab405c9c915

get(*request*, *l7_policy_id*)

Get a specific l7 policy.

If the param 'includeChildResources' is passed in as a truthy value, the details of all resources that exist under the l7 policy will be returned along with the l7 policy details.

http://localhost/api/lbaas/l7policies/cc758c90-3d98-4ea1-af44-aab405c9c915

put(*request*, *l7_policy_id*)

Edit a l7 policy as well as any resources below it.

url_regex = 'lbaas/l7policies/(?P<l7_policy_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**L7Rule**(***kwargs*)

Bases: View

API for retrieving a single l7 rule.

delete(*request*, *l7_rule_id*, *l7_policy_id*)

Delete a specific l7 rule.

get(*request*, *l7_rule_id*, *l7_policy_id*)

Get a specific l7 rule.

put(*request*, *l7_rule_id*, *l7_policy_id*)

Edit a specific l7 rule.

url_regex =
'lbaas/l7policies/(?P<l7_policy_id>[^/]+)/l7rules/(?P<l7_rule_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**L7Rules**(**kwargs)

Bases: View

API for load balancer l7 rules.

get(*request*, *l7_policy_id*)

List of l7 rules for the current project.

The listing result is an object with property "items".

post(*request*, *l7_policy_id*)

Create a new l7 rule.

Creates a new l7 rule as well as other optional resources such as l7 rules.

url_regex = 'lbaas/l7policies/(?P<l7_policy_id>[^/]+)/l7rules/$'

class octavia_dashboard.api.rest.lbaasv2.**Listener**(**kwargs)

Bases: View

API for retrieving, updating, and deleting a single listener.

delete(*request*, *listener_id*)

Delete a specific listener.

http://localhost/api/lbaas/listeners/cc758c90-3d98-4ea1-af44-aab405c9c915

get(*request*, *listener_id*)

Get a specific listener.

If the param 'includeChildResources' is passed in as a truthy value, the details of all resources that exist under the listener will be returned along with the listener details.

http://localhost/api/lbaas/listeners/cc758c90-3d98-4ea1-af44-aab405c9c915

put(*request*, *listener_id*)

Edit a listener as well as any resources below it.

url_regex = 'lbaas/listeners/(?P<listener_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**Listeners**(**kwargs)

Bases: View

API for load balancer listeners.

get(*request*)

List of listeners for the current project.

The listing result is an object with property "items".

post(*request*)

Create a new listener.

Creates a new listener as well as other optional resources such as a pool, members, and health monitor.

url_regex = 'lbaas/listeners/$'

class octavia_dashboard.api.rest.lbaasv2.**LoadBalancer**(***kwargs*)

Bases: View

API for retrieving, updating, and deleting a single load balancer.

delete(*request*, *loadbalancer_id*)

Delete a specific load balancer.

http://localhost/api/lbaas/loadbalancers/cc758c90-3d98-4ea1-af44-aab405c9c915

get(*request*, *loadbalancer_id*)

Get a specific load balancer.

http://localhost/api/lbaas/loadbalancers/cc758c90-3d98-4ea1-af44-aab405c9c915

put(*request*, *loadbalancer_id*)

Edit a load balancer.

url_regex = 'lbaas/loadbalancers/(?P<loadbalancer_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**LoadBalancers**(***kwargs*)

Bases: View

API for load balancers.

get(*request*)

List load balancers for current project.

The listing result is an object with property "items".

post(*request*)

Create a new load balancer.

Creates a new load balancer as well as other optional resources such as a listener, pool, monitor, etc.

url_regex = 'lbaas/loadbalancers/$'

class octavia_dashboard.api.rest.lbaasv2.**Member**(***kwargs*)

Bases: View

API for retrieving a single member.

delete(*request*, *member_id*, *pool_id*)

Delete a specific member belonging to a specific pool.

get(*request*, *member_id*, *pool_id*)

Get a specific member belonging to a specific pool.

put(*request*, *member_id*, *pool_id*)

Edit a pool member.

url_regex =
'lbaas/pools/(?P<pool_id>[^/]+)/members/(?P<member_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**Members**(***kwargs*)

Bases: View

API for load balancer members.

get(*request*, *pool_id*)

List of members for the current project.

The listing result is an object with property "items".

put(*request*, *pool_id*)

Update the list of members for the current project.

url_regex = 'lbaas/pools/(?P<pool_id>[^/]+)/members/$'

class octavia_dashboard.api.rest.lbaasv2.**Pool**(***kwargs*)

Bases: View

API for retrieving a single pool.

delete(*request*, *pool_id*)

Delete a specific pool.

http://localhost/api/lbaas/pools/cc758c90-3d98-4ea1-af44-aab405c9c915

get(*request*, *pool_id*)

Get a specific pool.

If the param 'includeChildResources' is passed in as a truthy value, the details of all resources that exist under the pool will be returned along with the pool details.

http://localhost/api/lbaas/pools/cc758c90-3d98-4ea1-af44-aab405c9c915

put(*request*, *pool_id*)

Edit a listener as well as any resources below it.

url_regex = 'lbaas/pools/(?P<pool_id>[^/]+)/$'

class octavia_dashboard.api.rest.lbaasv2.**Pools**(***kwargs*)

Bases: View

API for load balancer pools.

get(*request*)

List of pools for the current project.

The listing result is an object with property "items".

post(*request*)

Create a new pool.

Creates a new pool as well as other optional resources such as members and health monitor.

url_regex = 'lbaas/pools/$'

octavia_dashboard.api.rest.lbaasv2.**add_floating_ip_info**(*request*, *loadbalancers*)

Add floating IP address info to each load balancer.

octavia_dashboard.api.rest.lbaasv2.**add_member**(*request*, ***kwargs*)

Add a member to a pool.

octavia_dashboard.api.rest.lbaasv2.**create_flavor**(*request*, ***kwargs*)

Create a new flavor.

octavia_dashboard.api.rest.lbaasv2.**create_flavor_profile**(*request*, ***kwargs*)

Create a new flavor profile.

octavia_dashboard.api.rest.lbaasv2.**create_health_monitor**(*request*, ***kwargs*)

Create a new health monitor for a pool.

octavia_dashboard.api.rest.lbaasv2.**create_l7_policy**(*request*, ***kwargs*)

Create a new l7 policy.

octavia_dashboard.api.rest.lbaasv2.**create_l7_rule**(*request*, ***kwargs*)

Create a new l7 rule.

octavia_dashboard.api.rest.lbaasv2.**create_listener**(*request*, ***kwargs*)

Create a new listener.

octavia_dashboard.api.rest.lbaasv2.**create_loadbalancer**(*request*)

octavia_dashboard.api.rest.lbaasv2.**create_pool**(*request*, ***kwargs*)

Create a new pool.

octavia_dashboard.api.rest.lbaasv2.**get_members_to_add_remove**(*request_member_data*, *existing_members*)

octavia_dashboard.api.rest.lbaasv2.**health_monitor_get_load_balancer_id**(*conn*, *health_monitor_id*)

octavia_dashboard.api.rest.lbaasv2.**l7_policy_get_load_balancer_id**(*conn*, *l7_policy_id*)

octavia_dashboard.api.rest.lbaasv2.**listener_get_load_balancer_id**(*conn*, *listener_id*)

octavia_dashboard.api.rest.lbaasv2.**poll_loadbalancer_status**(*request*, *loadbalancer_id*, *callback*, *from_state='PENDING_UPDATE'*, *to_state='ACTIVE'*, *callback_kwargs=None*)

Poll for the status of the load balancer.

Polls for the status of the load balancer and calls a function when the status changes to a specified state.

Parameters

- **request** -- django request object
- **loadbalancer_id** -- id of the load balancer to poll
- **callback** -- function to call when polling is complete
- **from_state** -- initial expected state of the load balancer
- **to_state** -- state to check for
- **callback_kwargs** -- kwargs to pass into the callback function

octavia_dashboard.api.rest.lbaasv2.**pool_get_load_balancer_id**(*conn*, *pool_id*)

octavia_dashboard.api.rest.lbaasv2.**remove_member**(*request*, ***kwargs*)

Remove a member from the pool.

octavia_dashboard.api.rest.lbaasv2.**retry_on_conflict**(*conn*, *func*, **args*, *retry_timeout=120*, ***kwargs*)

octavia_dashboard.api.rest.lbaasv2.**update_flavor**(*request*, ***kwargs*)

Update a flavor.

octavia_dashboard.api.rest.lbaasv2.**update_flavor_profile**(*request*, ***kwargs*)

Update a flavor profile.

octavia_dashboard.api.rest.lbaasv2.**update_l7_policy**(*request*, ***kwargs*)

Update a l7 policy.

octavia_dashboard.api.rest.lbaasv2.**update_l7_rule**(*request*, ***kwargs*)

Update a l7 rule.

octavia_dashboard.api.rest.lbaasv2.**update_listener**(*request*, ***kwargs*)

Update a listener.

octavia_dashboard.api.rest.lbaasv2.**update_loadbalancer**(*request*, ***kwargs*)

Update a load balancer.

octavia_dashboard.api.rest.lbaasv2.**update_member_list**(*request*, ***kwargs*)

Update the list of members by adding or removing the necessary members.

octavia_dashboard.api.rest.lbaasv2.**update_monitor**(*request*, ***kwargs*)

Update a health monitor.

octavia_dashboard.api.rest.lbaasv2.**update_pool**(*request*, ***kwargs*)

Update a pool.

Module contents

This package holds the REST API that supports the Octavia dashboard Javascript code.

It is not intended to be used outside of Horizon, and makes no promises of stability or fitness for purpose outside of that scope.

It does not promise to adhere to the general OpenStack API Guidelines set out in https://wiki.openstack.org/wiki/APIChangeGuidelines.

Module contents

octavia_dashboard.dashboards package

Subpackages

octavia_dashboard.dashboards.project package

Subpackages

octavia_dashboard.dashboards.project.load_balancer package

Submodules

octavia_dashboard.dashboards.project.load_balancer.panel module

class octavia_dashboard.dashboards.project.load_balancer.panel.**NGLoadBalancers**

Bases: Panel

name = 'Load Balancers'

permissions = ('openstack.services.load-balancer',)

slug = 'load_balancer'

octavia_dashboard.dashboards.project.load_balancer.urls module

octavia_dashboard.dashboards.project.load_balancer.views module

class octavia_dashboard.dashboards.project.load_balancer.views.**IndexView**(***kwargs*)

Bases: HorizonTemplateView

page_title = 'Load Balancers'

template_name = 'project/load_balancer/index.html'

Module contents

Module contents

Module contents

Submodules

octavia_dashboard.sdk_connection module

octavia_dashboard.sdk_connection.**get_sdk_connection**(*request*)

Creates an SDK connection based on the request.

Parameters

request -- Django request object

Returns

SDK connection object

octavia_dashboard.version module

octavia_dashboard.version.**product_string**()

octavia_dashboard.version.**vendor_string**()

octavia_dashboard.version.**version_string_with_package**()

Module contents