Tuskar UI Documentation

Release Juno

Tuskar Team

May 05, 2015

Contents

Contents:

CHAPTER 1

Tuskar UI

Tuskar UI is a user interface for Tuskar, a management API for OpenStack deployments. It is a plugin for OpenStack Horizon.

1.1 High-Level Overview

Tuskar UI endeavours to be a stateless UI, relying on Tuskar API calls as much as possible. We use existing Horizon libraries and components where possible. If added libraries and components are needed, we will work with the OpenStack community to push those changes back into Horizon.

Interested in seeing Tuskar and Tuskar UI in action? Watch a demo!

1.2 Installation Guide

Use the Installation Guide to install Tuskar UI.

1.3 License

This project is licensed under the Apache License, version 2. More information can be found in the LICENSE file.

1.4 Contact Us

Join us on IRC (Internet Relay Chat):

```
Network: Freenode (irc.freenode.net/tuskar)
Channel: #tuskar
```

Or send an email to openstack-dev@lists.openstack.org.

CHAPTER 2

Installation instructions

2.1 Note

If you want to install and configure the entire TripleO + Tuskar + Tuskar UI stack, you can use the devtest installation guide.

Otherwise, you can use the installation instructions for Tuskar UI below.

2.2 Prerequisites

Installation prerequisites are:

1. A functional OpenStack installation. Horizon and Tuskar UI will connect to the Keystone service here. Keystone does *not* need to be on the same machine as your Tuskar UI interface, but its HTTP API must be accessible.
2. A functional Tuskar installation. Tuskar UI talks to Tuskar via an HTTP interface. It may, but does not have to, reside on the same machine as Tuskar UI, but it must be network accessible.

You may find the Tuskar install guide helpful.

2.3 Installing the packages

Tuskar UI is a Django app written in Python and has a few installation dependencies:

On a RHEL 6 system, you should install the following:

```
yum install git python-devel swig openssl-devel mysql-devel libxml2-devel libxslt-devel gcc gcc-c++
```

The above should work well for similar RPM-based distributions. For other distros or platforms, you will obviously need to convert as appropriate.

Then, you'll want to use the `easy_install` utility to set up a few other tools:

```
easy_install pip
easy_install nose
```

2.4 Install the management UI

Begin by cloning the Horizon and Tuskar UI repositories:

```
git clone git://github.com/openstack/horizon.git
git clone git://github.com/openstack/python-tuskarclient.git
git clone git://github.com/openstack/tuskar-ui.git
```

Go into `horizon` and install a virtual environment for your setup:

```
cd horizon
python tools/install_venv.py
```

Next, run `run_tests.sh` to have pip install Horizon dependencies:

```
./run_tests.sh
```

Set up your `local_settings.py` file:

```
cp openstack_dashboard/local/local_settings.py.example openstack_dashboard/local/local_settings.py
```

Open up the copied `local_settings.py` file in your preferred text editor. You will want to customize several settings:

- `OPENSTACK_HOST` should be configured with the hostname of your OpenStack server. Verify that the `OPENSTACK_KEYSTONE_URL` and `OPENSTACK_KEYSTONE_DEFAULT_ROLE` settings are correct for your environment. (They should be correct unless you modified your OpenStack server to change them.)

Install Tuskar UI with all dependencies in your virtual environment:

```
tools/with_venv.sh pip install -e ../python-tuskarclient/
tools/with_venv.sh pip install -e ../tuskar-ui/
```

And enable it in Horizon:

```
cp ../tuskar-ui/_50_tuskar.py.example openstack_dashboard/local/enabled/_50_tuskar.py
```

Then disable the other dashboards:

```
cp ../tuskar-ui/_10_admin.py.example openstack_dashboard/local/enabled/_10_admin.py
cp ../tuskar-ui/_20_project.py.example openstack_dashboard/local/enabled/_20_project.py
cp ../tuskar-ui/_30_identity.py.example openstack_dashboard/local/enabled/_30_identity.py
```

2.5 Starting the app

If everything has gone according to plan, you should be able to run:

```
tools/with_venv.sh ./manage.py runserver
```

and have the application start on port 8080. The Tuskar UI dashboard will be located at http://localhost:8080/infrastructure

If you wish to access it remotely (i.e., not just from localhost), you need to open port 8080 in iptables:

```
iptables -I INPUT -p tcp --dport 8080 -j ACCEPT
```

and launch the server with `0.0.0.0:8080` on the end:

```
tools/with_venv.sh ./manage.py runserver 0.0.0.0:8080
```

CHAPTER 3

User Guide

3.1 Nodes List File

To allow users to load a bunch of nodes at once, there is possibility to upload CSV file with given list of nodes. This file should be formatted as

```
driver,address,username,password/ssh key,mac addresses,cpu architecture,number of CPUs,available memo
```

Even if there is no all data available, we assume empty values for missing keys and try to parse everything, what is possible.

CHAPTER 4

Contributing

The code repository is located at OpenStack. Please go there if you want to check it out:

```
git clone https://github.com/openstack/tuskar-ui.git
```

The list of bugs and blueprints is on Launchpad:

https://launchpad.net/tuskar-ui

We use OpenStack's Gerrit for the code contributions:

https://review.openstack.org/#/q/status:open+project:openstack/tuskar-ui,n,z

and we follow the OpenStack Gerrit Workflow.

If you're interested in the code, here are some key places to start:

- tuskar_ui/api.py - This file contains all the API calls made to the Tuskar API (through python-tuskarclient).
- tuskar_ui/infrastructure - The Tuskar UI code is contained within this directory.

CHAPTER 5

Running tests

There are several ways to run tests for tuskar-ui.

Using `tox`:

> This is the easiest way to run tests. When run, tox installs dependencies, prepares the virtual python environment, then runs test commands. The gate tests in gerrit usually also use tox to run tests. For avaliable tox environments, see `tox.ini`.

By running `run_tests.sh`:

> Tests can also be run using the `run_tests.sh` script, to see available options, run it with the `--help` option. It handles preparing the virtual environment and executing tests, but in contrast with tox, it does not install all dependencies, e.g. `jshint` must be installed before running the jshint testcase.

Manual tests:

> To manually check tuskar-ui, it is possible to run a development server for tuskar-ui by running `run_tests.sh --runserver`.
>
> To run the server with the settings used by the test environment: `run_tests.sh --runserver 0.0.0.0:8000 --settings=tuskar_ui.test.settings`

CHAPTER 6

OpenStack Style Commandments

- Step 1: Read http://www.python.org/dev/peps/pep-0008/
- Step 2: Read http://www.python.org/dev/peps/pep-0008/ again
- Step 3: Read https://github.com/openstack-dev/hacking/blob/master/HACKING.rst

CHAPTER 7

Indices and tables

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