Spectrometer Documentation

Release 0.1.0

Tom Nadeau, Mohammad Hassan Zahraee, Thanh Ha, Vasu Srinivasan

Mar 22, 2017

Contents

Contents:

CHAPTER 1

Quick Start Guide

The Spectrometer project consists of two sub-projects, the `server` and `web`.

Server side is Python driven and provides the API to collect Git and Gerrit statistics for various OpenDaylight projects.

The web project is NodeJS/React based and provides the visualization by using the APIs provided by the server side.

In order to run the application, you need to install both `server` and `web` sub-projects.

This Quick Started Guide assumes you have Python3 and NodeJS 4.3 installed. To install NodeJS using NVM, see Web > Installation section below.

The Spectrometer project collects data from repositories located locally in your system.

Setup spectrometer-server

Installing spectrometer from pypi is simple and will get you the latest version that is released. Then create a config.py file in /etc/spectrometer/config.py (Example file can be found here)

```
pip install spectrometer
sudo mkdir /etc/spectrometer
sudo vi /etc/spectrometer/config.py
spectrometer server start
```

Verify that spectrometer-server is running by going to **http://localhost:5000**. You should see a Hello World page.

Setup spectrometer-web

Spectrometer Web is still in development so you will need to install it from Git at the time being as there is no package for it yet.

```
git clone https://git.opendaylight.org/gerrit/spectrometer.git
cd spectrometer/web
npm install
npm start
```

Goto **http://localhost:8000**

Testing the setup

By default the OpenDaylight project repositories will be mirrored every 5 minutes (300s), so if this is the first time starting you may have to wait until all repos are mirrored before you can exercise some of the apis.

Once the repos are mirrored you can try a few basic examples to make sure things are working properly:

Examples:

```
http://127.0.0.1:5000/gerrit/branches?project=controller
http://127.0.0.1:5000/gerrit/projects
http://127.0.0.1:5000/git/commits?project=integration/packaging
```

The full Rest APIs are documented here: https://opendaylight-spectrometer.readthedocs.io/en/latest/restapi.html

CHAPTER 2

User Guide

Spectrometer consists of 3 components:

- Spectrometer API Server (backend)
- Spectrometer Web Server (frontend)
- Spectrometer Report Tool

This guide will describe the uses of the 3 systems.

Spectrometer API Server

Production Deployment

When running in production the recommended way is to deploy with gunicorn.

```
gunicorn -b 0.0.0.0:5000 'spectrometer:run_app()'
```

If deploying behind a proxy under a sub-directory additional configuration is necessary for gunicorn application to operate correctly.

example-nginx:

```
location /api {
    proxy_pass         http://127.0.0.1:5000;
    proxy_redirect     http://127.0.0.1:5000/api/ http://$host/api/;

    proxy_set_header   Host             $host;
    proxy_set_header   X-Real-IP        $remote_addr;
    proxy_set_header   X-Forwarded-For  $proxy_add_x_forwarded_for;
    proxy_set_header   SCRIPT_NAME      /api;
}
```

Logging

Spectrometer logs to /var/log/spectrometer by default but that directory must be writeable by the spectrometer user.

```
sudo chown spectrometer /var/log/spectrometer
```

It is possible to override the default log directory by configuring the LOG_DIR parameter in config.py.

```
LOG_DIR = '/path/to/log/directory'
```

Spectrometer Web Server

TODO

Spectrometer Report Tool

The Spectrometer Report Tool can be used to generate reports between 2 reference points in time. Reference points are git commit hashs, branches, or tags. A project like OpenDaylight that tags projects with the same tag name for every release can use this tool to Generate release reports.

```
# spectrometer reporttool full <ref1> <ref2>
spectrometer reporttool --server-url=https://spectrometer.opendaylight.org/api full␣
↪release/beryllium-sr2 release/beryllium-sr1
```

CHAPTER 3

Project Info Specification

Spectrometer supports a PROJECT_INFO.yaml file placed in the root of a project repo. This file is used by spectrometer to parse meta information about the project including things like project description, project contact, committers irc, mailing lists, release names, etc...

```
# This file is used by Spectrometer to determine project meta information
# Please refer to the spec file located here:
# https://opendaylight-spectrometer.readthedocs.io/en/latest/project-info-spec.html

name: spectrometer
display-name: Spectrometer
creation-date: 2015-11-19
termination-date: n/a
description: |
    This is an example summary description of project

    After leaving a blank line in the description we can provide a longer
    more detailed description of the project.

    The details can be as many lines as necessary.
primary-contact: Firstname Lastname <first.last@example.com>
project-lead: Firstname Lastname <first.last@example.com>
categories:
    - application
    - community
    - documentation
    - extensions
    - kernel
    - library
    - protocols
    - services
committers:
    - Firstname Lastname <first.last@example.com>
    - Another Committer <another.committer@example.com>
# When Committers who have made significant contributions to OpenDaylight
# become inactive and thus no longer committers. This key can be used to
```

```
# acknowledge their huge contributions by appointing them to Committer
# Emeritus status.
committers-emeritus:
    - Firstname Lastname <first.last@example.com>
contributors:
    - Firstname Lastname <first.last@example.com>
    - Another Contributor <another.contributor@example.com>
wiki: https://wiki.example.org/project
irc: irc://irc.freenode.net/opendaylight-spectrometer
mailing-lists:
    - email: spectrometer-dev@lists.opendaylight.org
      archives: http://lists.opendaylight.org/pipermail/spectrometer-dev/
    - email: spectrometer-users@lists.opendaylight.org
      archives: http://lists.opendaylight.org/pipermail/spectrometer-users/
ci-server: https://jenkins.opendaylight.org
issue-tracker: https://bugs.opendaylight.org
static-analysis: https://sonar.opendaylight.org
repository: https://git.opendaylight.org/gerrit/#/admin/projects/spectrometer
meetings: |
    Free from text field for providing meeting information.
    It can be multiple lines long as necessary.
releases:
    - helium
    - lithium
    - beryllium
    - boron
```

Required fields:

- name
- creation_date
- description
- primary_contact
- project_lead

CHAPTER 4

Documentation Guide

This guide provides details on how to contribute to the documetantion of Spectrometer. The style guide we follow for documentation is the python documentation style guide. See:

https://docs.python.org/devguide/documenting.html

To build and review the documentation locally you can simply run tox and open the html via your favourite web browser.

```
tox -edocs
firefox .tox/docs/tmp/html/index.html
```

CHAPTER 5

Developer Guide

This doc provides details for developers who want to hack on spectrometer. If you have not done so already please refer to the *Quick Start Guide*.

Style Guide

We follow the Python PEP8 style guide. See: https://www.python.org/dev/peps/pep-0008/

For documentation we follow the Python Documentation Guide. See: https://docs.python.org/devguide/documenting.html

Spectrometer Server

Installing in Dev Mode

In development we want to install spectrometer so that we can modify the code and use it as if in production with changes taking effect immediately. We can achieve this using pip's editable install mode.

```
cd server  # From spectrometer repo root
pip install -e .
spectrometer server -c example-config/config.py start
```

Testing Code

We use tox to manage and run our unit tests. Simply run **tox** in the server directory to initiate the tests. If you don't have tox installed typically it is packaged as **python-tox** in most distros.

```
cd server/  # From spectrometer repo root
tox
```

Spectrometer Web

Installation

To install NodeJS in your system, use the Node Version Manager (NVM), which allows to co-exist multiple NodeJS versions in the same system.

If you already have NodeJS older versions (<= 0.12), it is strongly recommended to completely remove them and reinstall using NVM.

For Linux systems, you can do the following to remove NodeJS:

```
which node # Note down the path
sudo rm -r /path/bin/node /path/bin/npm /path/include/node /path/lib/node_modules ~/.
↪npm
```

Install NVM, NodeJS 4.3.x and NPM:

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.0/install.sh | bash
nvm install 4.3.1   # By default this installs npm 2.14.x
npm install npm -g  # This will upgrade npm to 3.7.x
```

Run spectrometer-web

cd web # From the root of the git repo npm install npm start

Goto `http://localhost:8000`

The web project is configured to hot-reload when any changes are made to the code. Most of the time the web browser should auto refresh, if not simply refresh the page.

UI Technology Stack

- NodeJS 4.3 - Bootstrapping and Universal (isomorphic) Javascript execution
- ExpressJS - Web-server-side bootstrap for UI
- ReactJS 0.14 - View Layer
- Redux - Data and State management (Flux pattern)
- Webpack - Build tool
- Babel - Asset compilation, ES6 Transpiler
- FormidableLabs VictoryChart - D3-based React components
- Redux Dev Tools - Tool that allows to track state management

Run spectrometer-web in Production

Production build does not have Devtools and hot reloading middleware. It also minifies scripts and css.

For Production build, execute the following commands:

```
npm run build
npm run start-prod
```

Run Test

Unit Tests are executed using Mocha and Chai assert libraries.

```
npm test
```

Roadmap

1. Dynamic loading of repositories as opposed to loading via config.json

Troubleshooting

Adding new repository

In order to add a new repository to collect statistics, you must make the following changes:

1. Create a soft link in ~/odl-spectrometer to the new repository
2. Edit the server/spectrometer/etc/repositories.yaml and specify the key and path to ~/odl-spectrometer/$repo
3. Edit the web/src/config.json add the project name in the list (this makes it appear in the dropdown)
4. Reload the web page
5. If reload web page does not work, restart python `` `python spectrometer-server` `` and web `` `npm start` ``)

CHAPTER 6

Rest API

Gerrit API

spectrometer.api.gerrit.**branches**()

Returns a list of branches in a given repository by querying Gerrit.

GET /gerrit/branches?param=<value>

Parameters **project** (*str*) – Project to query branches from. (required)

JSON:

```
{
  "branches": [
    {
      "ref": "refs/heads/stable/beryllium",
      "revision": "8f72284f3808328604bdff7f91a6999094f7c6d7"
    },
    ...
    ]
}
```

spectrometer.api.gerrit.**merged_changes**()

Returns a list of merged changes in a given repository by querying Gerrit.

GET /gerrit/changes?param=<value>

Parameters

- **project** (*str*) – Project to query changes from. (required)
- **branch** (*str*) – Branch to pull changes from. (default: master)

JSON:

```
{
  "changes": [
```

```
    {
      "_number": 37706,
      "branch": "master",
      "change_id": "I4168e023b77bfddbb6f72057e849925ba2dffa17",
      "created": "2016-04-18 02:42:33.000000000",
      "deletions": 0,
      "hashtags": [],
      "id": "spectrometer~master~I4168e023b77bfddbb6f72057e849925ba2dffa17",
      "insertions": 119,
      "owner": {
        "_account_id": 2759
      },
      "project": "spectrometer",
      "status": "MERGED",
      "subject": "Add API to return commits since ref",
      "submittable": false,
      "topic": "git-api",
      "updated": "2016-04-19 09:03:03.000000000"
    },
    ...
    ]
}
```


spectrometer.api.gerrit.**projects**()

Returns a list of projects by querying Gerrit.

GET /gerrit/projects

JSON:


```
{
  "projects": [
    "groupbasedpolicy",
    "spectrometer",
    "releng/autorelease",
    "snmp4sdn",
    "ovsdb",
    "nemo",
    ...
    ]
}
```


spectrometer.api.gerrit.**tags**()

Returns a list of tags in a given repository by querying Gerrit.

GET /gerrit/tags?param=<value>

Parameters **project** (*str*) – Project to query tags from. (required)

JSON:


```
{
  "tags": [
    {
      "message": "OpenDaylight Beryllium-SR1 release",
      "object": "f76cc0a12dc8f06dae3cedc31d06add72df8de5d",
      "ref": "refs/tags/release/beryllium-sr1",
      "revision": "8b92d614ee48b4fc5ba11c3f38c92dfa14d43655",
      "tagger": {
        "date": "2016-03-23 13:34:09.000000000",
```

```
        "email": "thanh.ha@linuxfoundation.org",
        "name": "Thanh Ha",
        "tz": -240
      }
    },
    ...
    ]
}
```

Git API

spectrometer.api.git.**branches**()

Returns a list of branches in a given repository.

GET /git/branches?param=<value>

Parameters **project** (*str*) – Project to query commits from. (required)

JSON:

```
{
  "branches": [
    "master",
    "stable/beryllium",
    "stable/helium",
    "stable/lithium",
    ...
  ]
}
```

spectrometer.api.git.**commits**()

Returns a list of commit messages in a repository.

GET /git/commits?param=<value>

Parameters

- **project** (*str*) – Project to query commits from. (required)
- **branch** (*str*) – Branch to pull commits from. (default: master)

JSON:

```
{
  "commits": [
    {
      "author": "Thanh Ha",
      "author_email": "thanh.ha@linuxfoundation.org",
      "author_tz_offset": 14400,
      "authored_date": 1460316386,
      "committed_date": 1460392605,
      "committer": "Thanh Ha",
      "committer_email": "thanh.ha@linuxfoundation.org",
      "committer_tz_offset": 14400,
      "hash": "1e409af62fd99413c5be86c5b43ad602a8cebc1e",
      "lines": {
        "deletions": 55,
        "files": 7,
```

```
        "insertions": 103,
        "lines": 158
      },
      "message": "Refactor Gerrit API into a Flask Blueprint..."
    },
    ...
  ]
}
```

Note:

date The date represented in seconds since epoch

tz_offset The seconds offset west of UTC.

spectrometer.api.git.**commits_since_ref**()

Returns a list of commits in branch until common parent of ref.

Searches Git for a common_parent between *ref1* and *ref2* and returns a the commit log of all the commits until the common parent excluding the common_parent commit itself.

GET /git/commits_since_ref?param=<value>

Parameters

- **project** (*str*) – Project to query commits from. (required)
- **ref1** (*str*) – Reference to get commit information from. (required)
- **ref2** (*str*) – Reference to start at until ref1. (required)

JSON:

```
{
  "commits": [
    {
      "author": "Thanh Ha",
      "author_email": "thanh.ha@linuxfoundation.org",
      "author_tz_offset": 14400,
      "authored_date": 1460316386,
      "committed_date": 1460392605,
      "committer": "Thanh Ha",
      "committer_email": "thanh.ha@linuxfoundation.org",
      "committer_tz_offset": 14400,
      "hash": "1e409af62fd99413c5be86c5b43ad602a8cebc1e",
      "lines": {
        "deletions": 55,
        "files": 7,
        "insertions": 103,
        "lines": 158
      },
      "message": "Refactor Gerrit API into a Flask Blueprint..."
    },
    ...
  ]
}
```

spectrometer.api.git.**project_info**()

Provides meta information on project.

Refer to the specfile located here: https://opendaylight-spectrometer.readthedocs.io/en/latest/project-info-spec.html

Python Module Index

S

Index