Contribution Guidelines
OPNFV Coding Guidelines

- OPNFV Coding Guidelines
  - OPNFV IP Policy needs to be signed off before contributing to OPNFV
  - Git commit message style
    - In a nutshell
    - Commit message content
    - Please submit small commits
    - General Code headers
      - C/C++/Java
      - Bash/Python
      - XML
      - Patch
      - Documentation
    - General Code Style
      - Java
      - Python
    - Notes on Copyright & Licence
      - License
      - Copyright
      - References
    - Git Repo Content Guidelines (draft)

NOTE: The OPNFV IP Policy needs to be signed off before contributing to OPNFV

Git commit message style

For Git commit messages we recommend following the OpenStack commit message recommendations. See: https://wiki.openstack.org/wiki/GitCommitMessages

In a nutshell

- Provide a brief description of the change in the first line.
- Insert a single blank line after the first line.
- Provide a detailed description of the change in the following lines. Use breaking paragraphs where needed.
- The first line should be limited to 50 characters; should be written in present tense; and should not end with a period.
- Subsequent lines should be wrapped at 72 characters.
- You should provide a JIRA issue link, using this format: JIRA: YARDSTICK-XXX
- At the end of the commit message, you should include the Change-Id and the Signed-off-by email.

Title of the message in present tense

Body of the message, describing the patch. Remember: less is not more; more is more.

JIRA: YARDSTICK-000

Change-Id: Iccd00000000000000000000000000000000000001
Signed-off-by: Name Surname <name.surname@company.com>

Commit message content

The commit message must contain all the information required to fully understand & review the patch for correctness. Less is not more. More is more.

Please submit small commits

Ensure there is only one "logical change" per commit. There are many reasons why this is an important rule:
- The least amount of code being changed, the quicker and easier it is to review and identify potential flaws.
- If a change is found to be flawed later, it may be necessary to revert the broken commit. This is much easier to do if there are not other unrelated code changes entangled with the original commit.
- When troubleshooting problems using Git's bisect capability; small well-defined changes will aid in isolating exactly where the code problem was introduced.
- When browsing history using Git's annotate/blame, small well-defined changes also aid in isolating exactly where a piece of code came from and why.

General Code headers

Headers must be applied to all contributed code. These should provide a copyright notice identifying the individual, and company where affiliated, of origin for the contribution and the license under which the contribution was made. In OPNFV we use the Apache 2.0 license, some templates for code headers follow.

You can also use the SPDX shorthand (vs. "http://www.apache.org/licenses/LICENSE-2.0") in the headers. The shorthand text to replace the URL is "SPDX-License-Identifier: Apache-2.0"

C/C++/Java

/*******************************************************************************
 * Copyright (c) 2017 <Company or Individual> and others.
 * All rights reserved. This program and the accompanying materials
 * are made available under the terms of the Apache License, Version 2.0
 * which accompanies this distribution, and is available at
 * http://www.apache.org/licenses/LICENSE-2.0
 */

Bash/Python

# Copyright (c) 2017 <Company or Individual> and others.
#
# All rights reserved. This program and the accompanying materials
# are made available under the terms of the Apache License, Version 2.0
# which accompanies this distribution, and is available at
# http://www.apache.org/licenses/LICENSE-2.0
#

XML

<!--
Copyright (c) 2017 <Company or Individual> and others.

All rights reserved. This program and the accompanying materials
are made available under the terms of the Apache License, Version 2.0
which accompanies this distribution, and is available at
http://www.apache.org/licenses/LICENSE-2.0
-->

Patch

: Copyright (c) 2017 <Company or Individual> and others.
: All rights reserved. This program and the accompanying materials
: are made available under the terms of the Apache License, Version 2.0
: which accompanies this distribution, and is available at
: http://www.apache.org/licenses/LICENSE-2.0

Documentation
Per [OPNFV IP Policy](https://www.opennetworking.org/opnfv), all documentation will be received and made available by the OPNFV Project under the Creative Commons Attribution 4.0 International License.

All documentation files need to be licensed using the creative commons license. The following example may be applied in the first lines of all contributed RST files:

```
.. This work is licensed under a Creative Commons Attribution 4.0 International License.
.. http://creativecommons.org/licenses/by/4.0
.. (c) <optionally add copywriters name>
```

The SPDX shorthand for the creative commons URL is "SPDX-License-Identifier: CC-BY-4.0". If using the SPDX shorthand the header would look like the following:

```
.. This work is licensed under a Creative Commons Attribution 4.0 International License.
.. SPDX-License-Identifier: CC-BY-4.0
.. (c) <optionally add copywriters name>
```

These lines will not be rendered in the html and pdf files.

**General Code Style**

Depends on the upstream project to which we intend to contribute to. If it's OPNFV specific 120 character line length would suffice. Yet, for OpenStack, a lot of projects enforce an 80 character length limit.

**Java**

In General we follow the Google Java Code Style Guide with a few exceptions.

See: [https://google-styleguide.googlecode.com/svn/trunk/javaguide.html](https://google-styleguide.googlecode.com/svn/trunk/javaguide.html)

- 120 character line length
- 72 or 80 chars for javadoc

**Python**

PEP8 is the Python standard that should be followed when coding any Python code with the following exceptions.

- 80 character line length

The usage of tools such as [flake8](https://flake8.pycodestyle.org/) is encouraged.

**Notes on Copyright & Licence**

In addition of the general code header mentioned above, the following recommendations apply.

**License**

- Each original source document (code and documentation, but excluding things like LICENSE and NOTICE files) MUST include a license header at the top (see headers above).
- License header does NOT need to be included in `__init__.py` files
- Third party files with non-apache v2.0 licenses should be put in "third_party" directory (preferably under the root of the projectc directory) to ease license scanning

**Copyright**

- As the code is version controlled, it is recommended NOT to include individual or entity authors in the headers (Apache recommendation), a global AUTHORS file MAY be generated based on git to list all the authors
- If you update a file, you could add the entity you represent (self or organization) to the list of Copyright holders. **However, you MUST NOT remove any listed Copyright headings.**

Following is an example:

```
# Copyright (c) 2016 <Company A> and others. => Existing copyright heading
#
# ABC Code is Copyright (c) 2017 <Company B> and others. => New copyright heading and "ABC Code" is a new/discrete code
```

**References**
Git Repo Content Guidelines (draft)

Best practices for what to include (or not) in commits to OPNFV git repos. The Anteater tool will flag and potentially block commits that violate some of the "DO NOT"s below. See the Anteater tool page for help on how obtain an exception for these best practices as needed.

- **DO commit**
  - Code source files in any source language
  - Plain text or RST documents
  - Images included in RST documents, tests, etc
  - Rich text documents as the source file for images included in RST documents etc

- **DO NOT commit**
  - Binary files other than images or rich text documents as described above, e.g.
    - Object libraries
    - Zipped files
    - Tar files
  - Copies of code or other items that can (and should) be referenced from an external source, e.g.
    - Code used by the project but not essential to be maintained in the project’s repos
    - Document generation tools