# Contributing Guidelines

Some basic conventions for contributing to this project.

## General

Please make sure that there aren't existing pull requests attempting to address the issue mentioned. Likewise, please check for issues related to update, as someone else may be working on the issue in a branch or fork.

- Please open a discussion in a new issue / existing issue to talk about the changes you'd like to bring

- Develop in a topic branch, not master/develop

When creating a new branch, prefix it with the \_type\_ of the change (see section \*\*Commit Message Format\*\* below), the associated opened issue number, a dash and some text describing the issue (using dash as a separator).

For example, if you work on a bugfix for the issue #361, you could name the branch `fix361-template-selection`.

## Issues open to contribution

Want to contribute but don't know where to start? Have a look at the issues labeled with the `good first issue` label: https://github.com/portainer/portainer/issues?q=is%3Aopen+is%3Aissue+label%3A%22good+first+issue%22

## Commit Message Format

Each commit message should include a \*\*type\*\*, a \*\*scope\*\* and a \*\*subject\*\*:

```

<type>(<scope>): <subject>

```

Lines should not exceed 100 characters. This allows the message to be easier to read on github as well as in various git tools and produces a nice, neat commit log ie:

```

#271 feat(containers): add exposed ports in the containers view

#270 fix(templates): fix a display issue in the templates view

#269 style(dashboard): update dashboard with new layout

```

### Type

Must be one of the following:

- \*\*feat\*\*: A new feature

- \*\*fix\*\*: A bug fix

- \*\*docs\*\*: Documentation only changes

- \*\*style\*\*: Changes that do not affect the meaning of the code (white-space, formatting, missing

semi-colons, etc)

- \*\*refactor\*\*: A code change that neither fixes a bug or adds a feature

- \*\*test\*\*: Adding missing tests

- \*\*chore\*\*: Changes to the build process or auxiliary tools and libraries such as documentation

generation

### Scope

The scope could be anything specifying place of the commit change. For example `networks`,

`containers`, `images` etc...

You can use the \*\*area\*\* label tag associated on the issue here (for `area/containers` use `containers` as a scope...)

### Subject

The subject contains succinct description of the change:

- use the imperative, present tense: "change" not "changed" nor "changes"

- don't capitalize first letter

- no dot (.) at the end

## Contribution process

Our contribution process is described below. Some of the steps can be visualized inside Github via specific `status/` labels, such as `status/1-functional-review` or `status/2-technical-review`.

### Bug report

![portainer\_bugreport\_workflow](https://user-images.githubusercontent.com/5485061/45727219-50190a00-bbf5-11e8-9fe8-3a563bb8d5d7.png)

### Feature request

The feature request process is similar to the bug report process but has an extra functional validation before the technical validation as well as a documentation validation before the testing phase.

![portainer\_featurerequest\_workflow](https://user-images.githubusercontent.com/5485061/45727229-5ad39f00-bbf5-11e8-9550-16ba66c50615.png)