Contribution Process

How to Contribute

Contributors to the GODOT community are required to complete the GODOT Community Contributors Agreement form.

The list of contributors is provided here.

If you want to contribute to GODOT Community Development there are many things you can do:

- use the software and report bugs in the issue tracker
- add examples to the documentation or the tutorials projects
- improve the documentation, adding better examples or clearer descriptions
- create a plugin extension and propose it for addition to the main libraries
- create a new application and make it available to the community

Changes to the GODOT and GODOTPY projects are only performed via ESA gitlab. If you have a suggestion for a change, raise an issue and the GODOT team can evaluate it.

Project Branches

Software contributions follow a trunk/master approach:

- Master: Is the single branch where new features / bug fixes are merged. CI/CD works with this branch
- Release: Developers don't work on release branches. In the case of a bug fix, you will create a branch from the trunk/master implement the fix and then merge into trunk/master first. Then based on the current situation, you can do another release from a new release branch which includes the fix or can cherry pick the fix to the current release branch.
- Feature / Hotfix: Branches created based on Master and merged into Master.

Fork and Branch

Software contributions may also be performed through a Fork and Branch model. Contributions are expected to be done through the following workflow:

- Fork a CoDev repository, and clone the fork to local
- Add a Git remote for the original repository (optional step)
- Create a feature branch in which changes are placed
- Make and commit changes to the new branch
- Push the branch into the created fork in CoDev
- Open a merge request from the new branch to the original CoDev repo, the development branch
- Clean after your merge request is merged

Please, read the following instructions for additional help:

https://blog.scottlowe.org/2015/01/27/using-fork-branch-git-workflow/

Code Reviews

For software contributions a Review-Then-Commit (RTC) approach is applicable:

- Develop the new feature or bug fix in a dedicated branch (consider also fork-and-branch)
- Create a merge request from your branch to the target branch (develop, release or hotfix)
- Your merge request will be reviewed by the core team (maintainers), and a discussion could be started on it. You could be asked to do some changes in order to facilitate consensus.
- Once the merge request is accepted, a maintainer will merge it into the corresponding protected branch