# How to Contribute

This project welcomes third-party code via GitHub pull requests.

You are welcome to propose and discuss enhancements using project [issues](https://github.com/IntelRealSense/librealsense/issues).

> \*\*Branching Policy\*\*:

> The `master` branch is considered stable, at all times.

> The `development` branch is the one where all contributions must be merged before being promoted to master.

> If you plan to propose a patch, please commit into the `development` branch, or its own feature branch.

We recommend enabling [travis-ci](https://travis-ci.org/) on your fork of `librealsense` to make sure the changes compile on all platforms and pass unit-tests.

In addition, please run `pr\_check.sh` and `api\_check.sh` under `scripts` directory. These scripts verify compliance with project's standards:

1. Every example / source file must refer to [LICENSE](https://github.com/IntelRealSense/librealsense/blob/master/LICENSE)

2. Every example / source file must include correct copyright notice

3. For indentation we are using spaces and not tabs

4. Line-endings must be Unix and not DOS style

5. Every API header file must be able to compile as the first included header (no implicit dependencies)

Most common issues can be automatically resolved by running `./pr\_check.sh --fix`

Please familirize yourself with the [Apache License 2.0](https://github.com/IntelRealSense/librealsense/blob/master/LICENSE) before contributing.

## Step-by-Step

1. Make sure you have `git` and `cmake` installed on your system. On Windows we recommend using [Git Extensions](https://github.com/gitextensions/gitextensions/releases) for git bash.

2. Run `git clone https://github.com/IntelRealSense/librealsense.git` and `cd librealsense`

3. To align with latest status of the development branch, run:

```

git fetch origin

git checkout development

git reset --hard origin/development

```

4. `git checkout -b name\_of\_your\_contribution` to create a dedicated branch

5. Make your changes to the local repository

6. Make sure your local git user is updated, or run `git config --global user.email "email@example.com"` and `git config --global user.user "user"` to set it up. This is the user & email that will appear in GitHub history.

7. `git add -p` to select the changes you wish to add

8. `git commit -m "Description of the change"`

9. Make sure you have a GitHub user and [fork librealsense](https://github.com/IntelRealSense/librealsense#fork-destination-box)

10. `git remote add fork https://github.com/username/librealsense.git` with your GitHub `username`

11. `git fetch fork`

12. `git push fork` to push `name\_of\_your\_contribution` branch to your fork

13. Go to your fork on GitHub at `https://github.com/username/librealsense`

14. Click the `New pull request` button

15. For `base` combo-box select `development`, since you want to submit a PR to that branch

16. For `compare` combo-box select `name\_of\_your\_contribution` with your commit

17. Review your changes and click `Create pull request`

18. Wait for all automated checks to pass

19. The PR will be approved / rejected after review from the team and the community

To continue to new change, goto step 3.

To return to your PR (in order to make more changes):

1. `git stash`

2. `git checkout name\_of\_your\_contribution`

3. Repeat items 5-8 from the previous list

4. `git push fork`

The pull request will be automatically updated

## Comment about the Wrappers

> It is very time consuming (and often impossible) for a single developer to test contributed functionality using all of the supported [wrappers](https://github.com/IntelRealSense/librealsense/tree/master/wrappers). There is no expectation of adding new functionality to all of the wrappers. One noteable exception is maintaining parity of public enumerations. Without strict maintanance it is easy for these lists to go out of sync and this can have serious runtime consequences.

For example, when adding new value to [`rs2\_option`](https://github.com/IntelRealSense/librealsense/blob/master/include/librealsense2/h/rs\_option.h) enum, please also add it to:

1. The list of Matlab options under [`wrappers/matlab/option.m`](https://github.com/IntelRealSense/librealsense/blob/master/wrappers/matlab/option.m#L3-L46)

2. The list of Node.js options [here](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/nodejs/index.js#L4661), [here](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/nodejs/index.js#L4927) and [here](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/nodejs/src/addon.cpp#L4629-L4692)

3. The list of options for [Unreal Engine](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/unrealengine4/Plugins/RealSense/Source/RealSense/Public/RealSenseTypes.h#L56-L118) integration

4. The list of options in the C# wrapper - [`wrappers/csharp/Intel.RealSense/Types/Enums/Option.cs`](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/csharp/Intel.RealSense/Types/Enums/Option.cs)

5. The list of Java options used for Android integration - [`wrappers/android/librealsense/src/main/java/com/intel/realsense/librealsense/Option.java`](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/android/librealsense/src/main/java/com/intel/realsense/librealsense/Option.java#L4-L64)

6. The list of options in the [python](https://github.com/IntelRealSense/librealsense/blob/v2.32.1/wrappers/python/pybackend.cpp#L102-L165) wrapper

Once all are updated [travis-ci](https://travis-ci.org/IntelRealSense/librealsense) will give clear indication that each of the wrappers is still passing compilation.