# How to contribute

Community contributions are an essential part of any open source project. The community has access to a large number of unique configurations which would

be extremely difficult for the core maintainers to reproduce. We want to keep it as easy as possible to contribute changes that get things working in your environment. There are a few guidelines that we need contributors to follow so that we can have a chance of keeping on top of things.

## Getting Started

If you are new to Dnn Development, you can start by watching this video playlist which explains how to setup the required tools, your development environment and submit pull requests for modules, this repository and the Persona Bar.

[How to make a pull requests playlist](https://www.youtube.com/playlist?list=PLIx1M8IdVvqZ0bnODGqJyxvONNPj5BzMP)

Please note that this playlist was recorded some time ago, if you encounter conflicting information, please rely on the most recent written documentation over what you see in those videos.

\* Make sure you have a [GitHub account](https://github.com/signup/free)

\* [Submit a ticket](https://github.com/dnnsoftware/Dnn.Platform/issues/new) for your issue, assuming one does not already exist. After review, if the pull request is accepted, we will mark the issue as resolved and assign it to a release so it can be QA'ed.

\* Clearly describe the issue including steps to reproduce when it is a bug.

\* Make sure you fill in the earliest version that you know has the issue.

\* We have included an issue template which helps make sure all of the relevant information is included with the issue report.

\* Add a comment to the ticket notifying the team of your intention to work on this item, including if known, an estimated time for Pull Request submission. This helps to limit duplication of works and allows the reviewers to plan for potential items incoming for a release.

## Fork the repository on GitHub

If this is your first time working with Dnn:Platform. You will need to fork the repository to get your system configured for local development.

\* Click fork on the main Dnn:Platform project

\* Clone your fork locally with `git clone https://github.com/[username]/Dnn.Platform.git` replacing [username] with your account information.

\* Add the upstream connection so you can rebase and update your fork with `git remote add upstream https://github.com/dnnsoftware/Dnn.Platform.git`

\* To update your fork to the latest, you can then run `git fetch upstream` followed by `git push`

## Making Changes

\* Create a topic branch from where you want to base your work.

\* This is usually the 'develop' branch.

\* Release branches should only be targeted in special cases and with approval from the approver group.

\* To quickly create a topic branch based on develop; `git checkout -b my\_contribution develop`

\* Make commits of logical units.

\* Check for unnecessary whitespace with `git diff --check` before committing.

\* When ready to publish your changes, you can with `git push -u origin my\_contribution`

\* Make sure your pull request description tags the GitHub issue ID, so it is clear what issue you have fixed.

\* Make sure your commit messages are in the proper format.

````

Short (72 chars or less) summary.

More detailed explanatory text. Wrap it to 72 characters. The blank line separating the summary from the body is critical (unless you omit

the body entirely).

Write your commit message in the imperative: "Fix bug" and not "Fixed bug" or "Fixes bug." This convention matches up with commit messages generated by commands like git merge and git revert.

Further paragraphs come after blank lines.

- Bullet points are okay, too.

- Typically a hyphen or asterisk is used for the bullet, followed by a

single space. Use a hanging indent.

Fix bug #123 #345

````

\* For bonus points run and add unit tests

\* Make sure you have added the necessary tests for your changes.

\* Run \_all\_ the tests to assure nothing else was accidentally broken.

> NOTE: Changes to methods that have been marked as [Obsolete] will typically not be accepted due to inherit risks associated with code scheduled for removal. If a critical bug/issue exists, please be sure to document & discuss using an RFC issue prior to submitting your contribution to ensure alignment with acceptance criteria.

## Making Trivial Changes

### Documentation

For changes of a trivial nature to comments and documentation, it is not

always necessary to create a new ticket in the issue tracker. In this case, it is

appropriate to start the first line of a commit with '(doc)' instead of

a ticket number.

````

Short (72 chars or less) summary starting with (doc).

The first line is a real-life imperative statement with '(doc)' in place of what would have been the ticket number in a non-documentation related commit. The body describes the nature of the new documentation or comments added.

````

## Submitting Changes

Once your change is completed you will want to follow the [Pull Request Process](https://github.com/dnnsoftware/Dnn.Platform/blob/develop/.github/PULL\_REQUEST\_PROCESS.md) to submit your changes for review & inclusion in the next release.

# Additional Resources

\* [General GitHub documentation](http://help.github.com/)

\* [GitHub pull request documentation](http://help.github.com/send-pull-requests/)