# Contributing

We've seen some great energy from folks around contributing to scriptcs and WE LOVE IT. scriptcs \*welcomes\* your contributions! In order to make it easy we've come up with the following model so that you can [Jump in](http://nikcodes.com/2013/05/10/new-contributor-jump-in/). The guidelines will help us all to operate in harmony.

We definitely want to soak up all that energy you have!

## The issue tracker and how you can contribute.

\* Bugs - For bugs, if you see one filed, just go fix it. If there is no bug filed, please file one first. You don't need to seek approval from core unless the bug starts to turn into a feature. \_Any bug fix which is a significant code contribution (say more than 10 lines of code), which introduces new APIs or which introduces breaking changes will be treated in the same way as we treat a feature (see below).\_

\* Features - By default all issues tagged as enhancements which we file will be fixed by the core team. However if you'd like to take it, please comment in the issue that you'd like to implement it so we can discuss before you put in a potentially wasted effort.

\* YOU TAKE IT - These are items that are important, but which we'd available for the community to take. One place we use this is for investigations and prototyping for features that would be really awesome, but which the core team just don't have bandwith for. As an example @dschenkelman [investigated](https://github.com/scriptcs/scriptcs/issues/68?source=cc) a VS debugging story. This lead to him ultimately implementing the feature, however don't feel pressure that you have to that. It's extremely valuable if you can just show us how it might be done.

### Other points

\* If an issue is marked as "Taken" then it is assigned to another member of the community.

\* To take an issue, reply in the comments "I'll take it" and wait for confirmation from a member of the core team.

## Discussing features

We'd like the design for all features to get socialized either via github issues or our [discussion group](https://groups.google.com/forum/?fromgroups#!forum/scriptcs). Please do this before you implement so that we can all get on the same page of the feature and not waste your time, which we really appreciate.

## Summary

\* Unassigned bug - Available for you

\* Assigned bug / Taken bug - unavailable

\* You take it issue - Available for you once it is confirmed.

\* Taken issue - Unavailable as someone already took it.

We really appreciate your help in taking scriptcs forward!

Also please see the discussion [here](https://github.com/scriptcs/scriptcs/issues/79) on which things are open for contribution.

## Workflow

\* We use the \*\*dev\*\* branch for active development. All pull requests should be made to this branch.

\* The master branch is only updated via pull requests from the dev branch.

\* Tests need to be provided for every bug/feature that is completed.

\* Please do not submit a PR unless without running all unit tests. If any tests are failing in the PR they will be rejected.

## Issue Management

We use some tags to manage issue priority.

\* \*\*P1\*\*: This is a big deal. It's either really awesome, terribly broken, or it's holding other awesome things up.

\* \*\*P2\*\*: This is kind of important and we're really looking forward to doing it, but there are more important things that need done first.

\* \*\*P3\*\*: We'd like to see these get implemented, but we're just not sure when we'll get to them. Want to take a shot at it? Go for it!

If the issue isn't marked with one of the above tags it's either still under discussion or not something we think we'll get to.

## Code Style

\* Indent with spaces (4) instead of tabs.

\* Prefix private instance field names with an underscore and be camel-cased. The `this` keyword should be avoided.

\* Use the `var` keyword unless the inferred type is not obvious.

\* Use the C# type aliases for types that have them (ex. `int` instead of `Int32`).

\* Use meaningful names (no hungarian notation).

\* Unit tests should follow the conventions detailed in [this blog post](http://haacked.com/archive/2012/01/02/structuring-unit-tests.aspx) by Phil Haack.

\* Unit tests assertions should be declared using the [Should Assertion Library](https://github.com/erichexter/Should) by Eric Hexter.