# Contributing

Contributions are always welcome! I only ask that you open an issue first so we can discuss the problem and solution. I just don't want you to waste any time headed in the wrong direction.

## Development setup

\* Clone this repo

\* Run `npm install -g gulp` and `npm install` in '/vscode-chrome-debug'

\* You may see an error if `bufferutil` or `utf-8-validate` fail to build. These native modules required by `ws` are optional and the adapter should work fine without them.

\* Run `gulp build`

## Developing in the vscode-chrome-debug-core module

Most of the code is actually in [this repo](https://github.com/Microsoft/vscode-chrome-debug-core) which is published in npm as `vscode-chrome-debug-core`. You can clone that repo separately to any directory and use `npm link` to test the extension with a modified version.

## Debugging

In VS Code, run the `launch as server` launch config - it will start the adapter as a server listening on port 4712. In your test app launch.json, include this flag at the top level: `"debugServer": "4712"`. Then you'll be able to debug the adapter in the first instance of VS Code, in its original TypeScript, using sourcemaps.

## Testing

There is a set of mocha tests which can be run with `gulp test` or with the `test` launch config. Also run `gulp tslint` to check your code against our tslint rules.

See the project under testapp/ for a bunch of test scenarios crammed onto one page.

## Naming

\* "Client": VS Code

\* "Target": The debuggee, which implements the Chrome Debug Protocol

\* "Server-mode": In the normal use-case, the extension does not run in server-mode. For debugging, you can run it as a debug server - see the 'Debugging' section above.

## Issue tags

\* "Bug": Something that should work is broken

\* "Enhancement": AKA feature request - adds new functionality

\* "Task": Something that needs to be done that doesn't really fix anything or add major functionality. Tests, engineering, documentation, etc.