# Contributing to the TSDoc project

## Building the @microsoft/tsdoc library

To build the project (\*\*standalone with NPM\*\*, if you are experimenting):

```shell

$ cd ./tsdoc

# Note: Be sure to specify "--no-package-lock" to avoid conflicts with Rush

$ npm install --no-package-lock

$ npm run build

```

\*\*-- OR --\*\*

To build the project (\*\*using Rush and Yarn\*\*, if you are contributing):

```shell

$ cd ./tsdoc

$ rush install

$ rush build

```

> For details about Rush, see \*\*"Submitting a PR"\*\* later on this page.

### Running the unit tests interactively

We use the [Jest](https://jestjs.io/) test runner. To run the TSDoc tests interactively (`jest --watch` scenario):

```shell

$ cd ./tsdoc

$ npm run watch

```

### To run all the unit tests from the command-line

Run all the unit tests once:

```shell

$ cd ./tsdoc

$ npm run build

$ npm run test

```

### Debugging the unit tests

The [./tsdoc/.vscode/launch.json](./tsdoc/.vscode/launch.json) file includes a

[Visual Studio Code](https://code.visualstudio.com/) configuration that makes debugging

easier. To debug a unit test:

1. Launch VS Code in the tsdoc subfolder (not the repository root):

```shell

$ cd ./tsdoc

$ code .

```

2. In the editor window, open a test file. For example, \*\*src/\_\_tests\_\_/ParsingBasicTests.test.ts\*\*.

3. Click the \*\*View --> Debug\*\* (CTRL+SHIFT+D)

4. From the DEBUG combo box, choose the "\*\*Jest Current File\*\*" debug configuration, and click the play button.

This will run only the unit tests in the currently opened file. (Use "\*\*Jest All\*\*" to run all tests in

the debugger.)

Jest is configured to run using a plugin that adds TypeScript support. You can set breakpoints in

the TypeScript source files, and the VS Code debugger will break on them.

## Submitting a PR

Before submitting your PR, you will need to install the Rush tool and build the monorepo.

> \*\*What's this Rush thing?\*\* Rush is Microsoft's build orchestrator that handles policy validation,

> change log management, installing (using [Yarn](https://yarnpkg.com/en/)), linking, building,

> and publishing. When it's time to publish the NPM package and deploy the TSDoc playground to the website,

> the automation system looks for its configuration in the [rush.json](./rush.json) file in this repo.

> To learn more about how to use Rush, please visit: https://rushjs.io/

1. [Install](https://rushjs.io/pages/developer/new\_developer/) the Rush software:

```shell

$ npm install -g @microsoft/rush

```

NOTE: If this command fails because your user account does not have permissions to

access NPM's global folder, you may need to

[fix your NPM configuration](https://docs.npmjs.com/getting-started/fixing-npm-permissions).

2. Install dependencies for all projects in the monorepo:

```shell

# Run this command in the folder where you cloned the TSDoc repo from GitHub

$ rush install

```

> IMPORTANT: After you run `rush install`, your repo will be in a "Rush-linked" state,

> with special symlinks in the node\_modules folders. DO NOT run `npm install` in this state.

> If you want to go back to working in standalone mode, first run `rush unlink && rush purge`.

3. Build and test all the projects in the monorepo:

```shell

$ rush build

```

You can also build just the \*\*@microsoft/tsdoc\*\* library like this:

```shell

$ cd ./tsdoc

$ npm run build

```

4. Manual testing: Before submitting your PR, you should also try running the

[/api-demo](./api-demo/) and [/playground](./playground) projects to make sure they

weren't broken by your change.

5. Change logs: If your PR modifies the published NPM package, you will need to write a

change entry for our [CHANGELOG.md](./tsdoc/CHANGELOG.md) change log. Please read the

"[recommended practices](https://rushjs.io/pages/best\_practices/change\_logs/)" for

authoring change logs.

```shell

$ rush change

# (The tool will ask you to write a sentence describing your change.)

```

The `rush change` command will create a file under the \*\*common/changes\*\* folder.

Add this file to Git and include in your pull request. Please see

[Everyday commands](https://rushjs.io/pages/developer/everyday\_commands/) for

more details about how these files are used.

## Contributor License Agreement (CLA)

This project welcomes contributions and suggestions. Most contributions require you to agree to a

Contributor License Agreement (CLA) declaring that you have the right to, and actually do, grant us

the rights to use your contribution. For details, visit https://cla.microsoft.com.

When you submit a pull request, a CLA-bot will automatically determine whether you need to provide

a CLA and decorate the PR appropriately (e.g., label, comment). Simply follow the instructions

provided by the bot. You will only need to do this once across all repos using our CLA.

This project has adopted the [Microsoft Open Source Code of Conduct](https://opensource.microsoft.com/codeofconduct/).

For more information see the [Code of Conduct FAQ](https://opensource.microsoft.com/codeofconduct/faq/) or

contact [opencode@microsoft.com](mailto:opencode@microsoft.com) with any additional questions or comments.