# Contributing to chartist-js

- [Issues and Bugs](#issue)

- [Submission Guidelines](#submit)

- [Coding Conventions](#conventions)

## <a name="issue"></a> Found an Issue?

If you find a bug in the source code or a mistake in the documentation, you can help us by

submitting an issue to our [GitHub Repository][github]. Even better you can submit a Pull Request

with a fix.

## Pre-requisites

You will need the following to run a local development enviroment.

- Node.js & npm

- Bower (`sudo npm install bower -g`)

- Grunt (`sudo npm install grunt-cli -g`)

- Text editor of your choice

## How to Run a Local Distribution

1. `cd` into your local copy of the repository.

2. Run `npm install` to install dependencies located in `package.json`.

3. Run `bower install` to install bower dependencies.

5. Run `grunt preview` to start the watch task, and the web server should automatically open. Congrats, you should now be able to see your local copy of the demo site.

## <a name="submit"></a> Submission Guidelines

If you are creating a Pull Request, fork the repository and make any changes on the `develop` branch.

### <a name="conventions"></a> Conventions

Check out the [Coding Style document](CODINGSTYLE.md)

### Grunt

We have five grunt tasks:

1. `grunt build` - Combines the scripts and creates the library for distribution

2. `grunt public` - Creates the distribution of the example / demo site which is used as visual development help of the charts but also serves as the documentation site / gh-pages.

3. `grunt dev` - Starts watch with livereload that is executing the same things as the site build default task but for live development.

4. `grunt preview` - Executes a dist and serves the directory statically in order to serve with the production example / demo site.

5. `grunt test` - Executes jasmine tests separately, although we have a very big lack of tests.

`dist` should \*\*not\*\* be included in any Pull Requests. So please ensure that code is not being committed as part of the Pull Request.

### Documentation

- Everything is already in place and in the `sitedist` there is a `apidoc` folder generated by [doxication](https://github.com/gionkunz/grunt-doxication) generator that uses JSDoc like comments to generate documentation meta files. Always use proper JSDoc comments when documenting methods and API interfaces. Also assign documentation blocks using @memberof to the virtual module they belong to.

- The site documentation is built with [Assemble.io](http://assemble.io/). Generally a component based approach should be followed where there are already Handlebar partials / helpers in order to create whole sites based on components that can be specified by type and with their data in yaml files.

### Important missing stuff

1. Jasmine Tests!

2. Documentation: JSDoc, Getting started documentation and landing page

3. Better accessibility using ARIA and other optimizations

4. Better interfaces to the library (i.e. jQuery with data-\* attributes for configuration), Angular.js directive etc.

5. Richer Sass / CSS framework

6. Other charts types (spider etc.)