Submitting Issues

=================

If you are submitting a bug, please create a [jsfiddle](http://jsfiddle.net/) demonstrating the issue.

Contributing code

=================

To contribute, fork the library and install gulp and dependencies. You need [node](http://nodejs.org/); use [nvm](https://github.com/creationix/nvm) or [nenv](https://github.com/ryuone/nenv) to install it.

```shell

git clone https://github.com/dalelotts/angular-bootstrap-datetimepicker.git

cd angular-bootstrap-datetimepicker

npm install

git checkout -b my-fix-branch develop # all patches against develop branch, please!

npm test # this runs lint, complexity checks, and unit tests.

```

Very important notes

====================

\* \*\*Pull pull requests to the `master` branch will be closed.\*\* Please submit all pull requests to the `develop` branch.

\* \*\*Pull requests will not be merged without unit tests.\*\*

\* \*\*Do not include the minified files in your pull request.\*\*

\* \*\*Have good tests. If you don't have tests for every line and branch in your changes, I won't accept the PR.

\* \*\*If your PR fails the CI build, I won't look at it.

Gulp tasks

===========

We use Gulp for managing the build. Here are some useful Gulp tasks:

\* `npm test` Runs all tests (webpack, commonjs, etc) checks the coding style, lints the code, calculates complexity, runs all tests, and enforces code coverage. You should make sure you do this before submitting a PR.

\* `gulp` The default task checks the coding style, lints the code, calculates complexity, runs the tests (not webpack or commonJS tests), and enforces code coverage.

\* `gulp scss` Generates the css file from the source scss.

\* `gulp css-lint` Lint the css files after generating.

\* `gulp templatecache` Generates src/js/datetimepicker.templates.js. You must re-add the IIFE around the generated code after it is generated (PR to fix this would be apprecaited)

# Contributing to angular-bootstrap-datetimepicker

We'd love for you to contribute to our source code and to make angular-bootstrap-datetimepicker even better than it is

today! Here are the guidelines we'd like you to follow:

- [Question or Problem?](#question)

- [Issues and Bugs](#issue)

- [Feature Requests](#feature)

- [Submission Guidelines](#submit)

- [Coding Rules](#rules)

- [Commit Message Guidelines](#commit)

- [Signing the CLA](#cla)

- [Further Info](#info)

## <a name="question"></a> Got a Question or Problem?

If you have questions about how to use angular-bootstrap-datetimepicker, please direct these to the [Gitter Chat Group][gitter].

## <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][issues]. Even better you can submit a Pull Request

with a fix.

\*\*Localization Issues:\*\* This directive uses the [Moment][momentI18N] for all localization.

This means that any localization issues should be reproted to Moment.

\*\*Please see the [Submission Guidelines](#submit) below.\*\*

## <a name="feature"></a> Want a Feature?

You can request a new feature by submitting an issue to our [GitHub Repository][github]. If you

would like to implement a new feature then consider what kind of change it is:

\* \*\*Major Changes\*\* that you wish to contribute to the project should be discussed first on [Gitter Chat][gitter]

so that we can better coordinate our efforts, prevent duplication of work, and help you to craft the

change so that it is successfully accepted into the project.

\* \*\*Small Changes\*\* can be crafted and submitted to the [GitHub Repository][github] as a Pull

Request.

## <a name="docs"></a> Want a Doc Fix?

If you want to help improve the docs, submit a PR.

When naming the commit, it is advised to follow the commit message

guidelines below, by starting the commit message with \*\*docs\*\* and

reference the filename and follows the \*\*[Commit Message Guidelines](#commit)\*\* outlined below.

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

### Submitting an Issue

Before you submit your issue search the archive, maybe your question was already answered.

If your issue appears to be a bug, and hasn't been reported, open a new issue. Help us to maximize

the effort we can spend fixing issues and adding new features, by not reporting duplicate issues.

Providing the following information will increase the chances of your issue being dealt with

quickly:

\* \*\*Overview of the Issue\*\* - if an error is being thrown a non-minified stack trace helps

\* \*\*Motivation for or Use Case\*\* - explain why this is a bug for you

\* \*\*Angular Version(s)\*\* - is it a regression?

\* \*\*Browsers and Operating System\*\* - is this a problem with all browsers or only specific ones?

\* \*\*Reproduce the Error\*\* - provide a live example (using [Plunker][plunker] or

[JSFiddle][jsfiddle]) or an unambiguous set of steps.

\* \*\*Related Issues\*\* - has a similar issue been reported before?

\* \*\*Suggest a Fix\*\* - if you can't fix the bug yourself, perhaps you can point to what might be

causing the problem (line of code or commit)

Here is a great example of a well defined issue: https://github.com/angular/angular.js/issues/5069

\*\*If you get help, help others. Good karma rulez!\*\*

### Submitting a Pull Request

Before you submit your pull request consider the following guidelines:

\* Search [GitHub][pulls] for an open or closed Pull Request

that relates to your submission. You don't want to duplicate effort.

\* Make your changes in a new git branch:

```shell

git checkout -b my-fix-branch master

```

\* Create your patch, \*\*including appropriate test cases\*\*.

\* Run the full test suite, `npm test`, and ensure that all tests pass.

\* Commit your changes using a descriptive commit message that follows our

[commit message conventions](#commit). Adherence to the [commit message conventions](#commit) is required,

because release notes are automatically generated from these messages.

```shell

git commit -a

```

Note: the optional commit `-a` command line option will automatically "add" and "rm" edited files.

\* Build your changes locally to ensure all the tests pass:

```shell

npm test

```

\* Push your branch to your fork on GitHub:

```shell

git push origin my-fix-branch

```

In GitHub, send a pull request to `angular-bootstrap-datetimepicker:develop`.

If we suggest changes, then:

\* Make the required updates.

\* Re-run the test suite to ensure tests are still passing.

\* Commit your changes to your branch (e.g. `my-fix-branch`).

\* Push the changes to your GitHub repository (this will update your Pull Request).

If the PR gets too outdated we may ask you to rebase and force push to update the PR:

```shell

git rebase master -i

git push origin my-fix-branch -f

```

\_WARNING: Squashing or reverting commits and force-pushing thereafter may remove GitHub comments

on code that were previously made by you or others in your commits. Avoid any form of rebasing

unless necessary.\_

That's it! Thank you for your contribution!

#### After your pull request is merged

After your pull request is merged, you can safely delete your branch and pull the changes

from the main (upstream) repository:

\* Delete the remote branch on GitHub either through the GitHub web UI or your local shell as follows:

```shell

git push origin --delete my-fix-branch

```

\* Check out the master branch:

```shell

git checkout master -f

```

\* Delete the local branch:

```shell

git branch -D my-fix-branch

```

\* Update your master with the latest upstream version:

```shell

git pull --ff upstream master

```

## <a name="rules"></a> Coding Rules

To ensure consistency throughout the source code, keep these rules in mind as you are working:

\* All features or bug fixes \*\*must be tested\*\* by one or more test.

## <a name="commit"></a> Git Commit Guidelines

We have very precise rules over how our git commit messages can be formatted. This leads to \*\*more

readable messages\*\* that are easy to follow when looking through the \*\*project history\*\*. But also,

we use the git commit messages to \*\*generate the angular-bootstrap-datetimepicker change log\*\*.

The commit message formatting can be added using a typical git workflow or through the use of a CLI wizard ([Commitizen](https://github.com/commitizen/cz-cli)). To use the wizard, run `npm run commit` in your terminal after staging your changes in git.

### Commit Message Format

Each commit message consists of a \*\*header\*\*, a \*\*body\*\* and a \*\*footer\*\*. The header has a special

format that includes a \*\*type\*\*, a \*\*scope\*\* and a \*\*subject\*\*:

```

<type>(<scope>): <subject>

<BLANK LINE>

<body>

<BLANK LINE>

<footer>

```

The \*\*header\*\* is mandatory and the \*\*scope\*\* of the header is optional.

Any line of the commit message cannot be longer 100 characters! This allows the message to be easier

to read on GitHub as well as in various git tools.

### Revert

If the commit reverts a previous commit, it should begin with `revert: `, followed by the header of the reverted commit. In the body it should say: `This reverts commit <hash>.`, where the hash is the SHA of the commit being reverted.

### Type

Must be one of the following:

\* \*\*feat\*\*: A new feature

\* \*\*fix\*\*: A bug fix

\* \*\*docs\*\*: Documentation only changes

\* \*\*style\*\*: Changes that do not affect the meaning of the code (white-space, formatting, missing

semi-colons, etc)

\* \*\*refactor\*\*: A code change that neither fixes a bug nor adds a feature

\* \*\*perf\*\*: A code change that improves performance

\* \*\*test\*\*: Adding missing tests

\* \*\*chore\*\*: Changes to the build process or auxiliary tools and libraries such as documentation

generation

### Scope

The scope could be anything specifying place of the commit change. For example `DateObject`,

`configuration`, `yearView`, `monthView`, `dayView`, `hourVuew`, `minuteView`, `templates`, etc...

### Subject

The subject contains succinct description of the change:

\* use the imperative, present tense: "change" not "changed" nor "changes"

\* don't capitalize first letter

\* no dot (.) at the end

### Body

Just as in the \*\*subject\*\*, use the imperative, present tense: "change" not "changed" nor "changes".

The body should include the motivation for the change and contrast this with previous behavior.

### Footer

The footer should contain any information about \*\*Breaking Changes\*\* and is also the place to

reference GitHub issues that this commit \*\*Closes\*\*.

\*\*Breaking Changes\*\* should start with the word `BREAKING CHANGE:` with a space or two newlines. The rest of the commit message is then used for this.

[github]: https://github.com/dalelotts/angular-bootstrap-datetimepicker

[gitter]: https://gitter.im/dalelotts/angular-bootstrap-datetimepicker

[issues]: https://github.com/dalelotts/angular-bootstrap-datetimepicker/issues

[pulls]: https://github.com/dalelotts/angular-bootstrap-datetimepicker/pulls

[momentI18N]: http://momentjs.com/docs/#/i18n/

[jsfiddle]: http://jsfiddle.net/

[plunker]: http://plnkr.co/edit

[unit-testing]: https://docs.angularjs.org/guide/unit-testing