# Contributing

First of all, \*\*thank you\*\* for contributing, \*\*you are awesome\*\*!

This project uses the fork & pull model of development. This means that in order to contribute

you need to submit a [pull request](https://help.github.com/articles/using-pull-requests/).

For relatively small features, improvements and bug fixes ([example](https://github.com/sstalle/php7cc/commit/a9f40a363fab2b24506465f8849a82cb3542739a)),

you can submit pull requests without prior discussion. If you are planning on doing something that requires

a lot of changes and/or big refactoring ([example](https://github.com/sstalle/php7cc/commit/600f0f9848af1f5ab631114304e0683d512f532b)),

please open an issue first so it can be thoroughly considered and examined.

Here are a few rules to follow in order to ease code reviews and discussions before

maintainers accept and merge your work:

\* [Follow the coding standards](#coding-standards)

\* [Run and update the tests](#running-and-updating-test-suite)

\* [Document your work](#documenting-your-work)

Please [rebase your branch](http://git-scm.com/book/en/Git-Branching-Rebasing)

before submitting your Pull Request. One may ask you to [squash your

commits](http://gitready.com/advanced/2009/02/10/squashing-commits-with-rebase.html)

too. This is used to "clean" your Pull Request before merging it (we don't want

commits such as `fix tests`, `fix 2`, `fix 3`, etc.).

## Coding standards

You MUST follow the [PSR-1](http://www.php-fig.org/psr/1/),

[PSR-2](http://www.php-fig.org/psr/2/) and

[Symfony Coding Standard](http://symfony.com/doc/current/contributing/code/standards.html)

(the only exception is that you must add single spaces around the concatenation operator).

If you don't know about any of them, you should really read the recommendations.

To fix your code according to the project standards, you can run

[PHP-CS-Fixer tool](http://cs.sensiolabs.org/) before commit:

```bash

composer cs

```

## Running and updating test suite

\* You MUST run the test suite.

\* You MUST write tests for PHP 7 compatibility errors.

\* You SHOULD write (or update) unit tests for any other non-trivial functionality.

Test suite can be run using the following command:

```bash

composer test

```

In most cases you should not write unit tests for compatibility violation checking visitors (like the ones found

in ```src/NodeVisitor```). To test them, you should create a subfolder in ```test/resource```

folder and put a ```.test``` file in it. ```.test``` files have multiple sections separated by

`-----`:

1. First section is the description of the test suite. It can also contain PHP version constraint.

2. Second section is the php code to be tested. It must be syntactically correct, unless it is an expression

or a statement that had been correct in PHP 5 but is no longer correct in PHP 7.

3. Third section is a newline separated array of messages and errors that

should be emitted for the code from the previous section. Errors are instances of

\Exception and \ParseException that are thrown during the checks. If there should be no messages

and no errors, just leave a blank like in this section. Please keep in mind that test suites are

not isolated, so you may get messages from other checkers in your test suite.

Second and third sections can be repeated one or more times.

Some tests require a particular version of PHP. For example, the `yield` keyword

had been introduced in PHP 5.5, and tests containing it cannot be run on the lower versions.

To specify a version constraint for the test suite, add a new line of the following format

to the first section:

```

PHP <operator><version>

```

Operator is one of the operators supported by `version\_compare` function. Multiple space separated

constraints can be specified.

## Documenting your work

You SHOULD write documentation for the code you add.

Also, please, write [commit messages that make

sense](http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html).

While creating your Pull Request on GitHub, you MUST write a description

which gives the context and/or explains why you are creating it.

Thank you!