# Contribute to Doctrine

Thank you for contributing to Doctrine!

Before we can merge your Pull-Request here are some guidelines that you need to follow.

These guidelines exist not to annoy you, but to keep the code base clean,

unified and future proof.

## We only accept PRs to "master"

Our branching strategy is "everything to master first", even

bugfixes and we then merge them into the stable branches. You should only

open pull requests against the master branch. Otherwise we cannot accept the PR.

There is one exception to the rule, when we merged a bug into some stable branches

we do occasionally accept pull requests that merge the same bug fix into earlier

branches.

## Coding Standard

We use PSR-1 and PSR-2:

\* https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-1-basic-coding-standard.md

\* https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-2-coding-style-guide.md

with some exceptions/differences:

\* Keep the nesting of control structures per method as small as possible

\* Align equals (=) signs

\* Add spaces between assignment, control and return statements

\* Prefer early exit over nesting conditions

\* Add spaces around a negation if condition ``if ( ! $cond)``

## Unit-Tests

Please try to add a test for your pull-request.

\* If you want to fix a bug or provide a reproduce case, create a test file in

``tests/Doctrine/Tests/ORM/Functional/Ticket`` with the name of the ticket,

``DDC1234Test.php`` for example.

\* If you want to contribute new functionality add unit- or functional tests

depending on the scope of the feature.

You can run the unit-tests by calling ``vendor/bin/phpunit`` from the root of the project.

It will run all the tests with an in memory SQLite database.

In order to do that, you will need a fresh copy of the ORM, and you

will have to run a composer installation in the project:

```sh

git clone git@github.com:doctrine/orm.git

cd orm

curl -sS https://getcomposer.org/installer | php --

./composer.phar install

```

To run the testsuite against another database, copy the ``phpunit.xml.dist``

to for example ``mysql.phpunit.xml`` and edit the parameters. You can

take a look at the ``tests/travis`` folder for some examples. Then run:

vendor/bin/phpunit -c mysql.phpunit.xml

If you do not provide these parameters, the test suite will use an in-memory

sqlite database.

Tips for creating unit tests:

1. If you put a test into the `Ticket` namespace as described above, put the testcase and all entities into the same class.

See `https://github.com/doctrine/orm/tree/master/tests/Doctrine/Tests/ORM/Functional/Ticket/DDC2306Test.php` for an

example.

## Travis

We automatically run your pull request through [Travis CI](http://www.travis-ci.org)

against SQLite, MySQL and PostgreSQL. If you break the tests, we cannot merge your code,

so please make sure that your code is working before opening up a Pull-Request.

## Getting merged

Please allow us time to review your pull requests. We will give our best to review

everything as fast as possible, but cannot always live up to our own expectations.

Thank you very much again for your contribution!