# MathPHP Contributing Guide

Thank you in your interest in contributing to make Math PHP even better!

## License

MathPHP is licensed under the MIT License.

[Read the license](https://github.com/markrogoyski/math-php/blob/master/LICENSE.txt).

## Unit Tests

All Math PHP code is tested with PHPUnit. The goal is beyond 100% test coverage. For code to be accepted into MathPHP it \*\*must\*\* have unit tests.

When writing tests, the preferred pattern is to make a generic test that uses a data provider. Then, provide many sample inputs and outputs to thoroughly test the function.

Things to test:

\* Normal expected use cases (many of these)

\* Edge cases

\* 0

\* 1

\* negative numbers

\* small inputs

\* large inputs

\* empty input

\* etc.

Document where test data came from. If test data is from a known definitive source, provide a link to the source. If the test data was created from an online mathematical calculator or cross referenced with another mathematical library such as R or Excel, document the source.

In addition to testing each specific function, it is a MathPHP testing standard to go beyond 100% code coverage by also testing mathematical axioms that make use of functions. Create unit tests that test axioms related to the functions you are working on.

Learn more about [PHPUnit](http://www.phpunit.de/).

## Coding Standards

All code must follow the Math PHP [coding standards](https://github.com/markrogoyski/math-php/wiki/Coding-Standards).

## Project Organization

```

src/ <- code goes here

tests/ <- unit tests go here

```

Namespaces designate different fields of math, and within each namespace may be further designations within the field. Code should go in the appropriate namespace. A new namespace may be necessary if it is a new field of math not currently implemented in MathPHP.

## Git Workflow

Math PHP is developed using Gitflow.

The basic idea is that the develop branch is the main public branch, and every commit to develop is a possible release candidate. Do your development in a feature branch and merge to develop when ready to submit a pull request. Master branch is only for releases merged from develop.

[Learn more about Gitflow.](http://nvie.com/posts/a-successful-git-branching-model/)