# Contributing to ZipStream-PHP

## Welcome!

We look forward to your contributions! Here are some examples how you can contribute:

* [Report a bug](https://github.com/maennchen/ZipStream-PHP/issues/new?labels=bug&template=BUG.md)
* [Propose a new feature](https://github.com/maennchen/ZipStream-PHP/issues/new?labels=enhancement&template=FEATURE.md)
* [Send a pull request](https://github.com/maennchen/ZipStream-PHP/pulls)

## We have a Code of Conduct

Please note that this project is released with a [Contributor Code of Conduct](http://code_of_conduct.md). By participating in this project you agree to abide by its terms.

## Any contributions you make will be under the MIT License

When you submit code changes, your submissions are understood to be under the same [MIT License](https://github.com/maennchen/ZipStream-PHP/blob/main/LICENSE) that covers the project. By contributing to this project, you agree that your contributions will be licensed under its MIT License.

## Write bug reports with detail, background, and sample code

In your bug report, please provide the following:

* A quick summary and/or background
* Steps to reproduce
  + Be specific!
  + Give sample code if you can.
* What you expected would happen
* What actually happens
* Notes (possibly including why you think this might be happening, or stuff you
* tried that didn't work)

Please do not report a bug for a version of ZIPStream-PHP that is no longer supported (< 3.0.0). Please do not report a bug if you are using a version of PHP that is not supported by the version of ZipStream-PHP you are using.

Please post code and output as text ([using proper markup](https://guides.github.com/features/mastering-markdown/)). Do not post screenshots of code or output.

Please include the output of composer info | sort.

## Workflow for Pull Requests

1. Fork the repository.
2. Create your branch from main if you plan to implement new functionality or change existing code significantly; create your branch from the oldest branch that is affected by the bug if you plan to fix a bug.
3. Implement your change and add tests for it.
4. Ensure the test suite passes.
5. Ensure the code complies with our coding guidelines (see below).
6. Send that pull request!

Please make sure you have [set up your user name and email address](https://git-scm.com/book/en/v2/Getting-Started-First-Time-Git-Setup) for use with Git. Strings such as silly nick name <root@localhost> look really stupid in the commit history of a project.

We encourage you to [sign your Git commits with your GPG key](https://docs.github.com/en/github/authenticating-to-github/signing-commits).

Pull requests for new features must be based on the main branch.

We are trying to keep backwards compatibility breaks in ZipStream-PHP to a minimum. Please take this into account when proposing changes.

Due to time constraints, we are not always able to respond as quickly as we would like. Please do not take delays personal and feel free to remind us if you feel that we forgot to respond.

## Coding Guidelines

This project comes with a configuration file (located at /psalm.yml in the repository) that you can use to perform static analysis (with a focus on type checking):

$ .composer run test:lint

This project comes with a configuration file (located at /.php-cs-fixer.dist.php in the repository) that you can use to (re)format your source code for compliance with this project's coding guidelines:

$ composer run format

Please understand that we will not accept a pull request when its changes violate this project's coding guidelines.

## Using ZipStream-PHP from a Git checkout

The following commands can be used to perform the initial checkout of ZipStream-PHP:

$ git clone git@github.com:maennchen/ZipStream-PHP.git

$ cd ZipStream-PHP

Install ZipStream-PHP's dependencies using [Composer](https://getcomposer.org/):

$ composer install

$ composer run install:tools # Install phpDocumentor using phive

## Running ZipStream-PHP's test suite

After following the steps shown above, ZipStream-PHP's test suite is run like this:

$ composer run test:unit

There's some slow tests in the test suite that test the handling of big files in the archives. To skip them use the following command instead:

$ composer run test:unit:fast

## Generating ZipStream-PHP Documentation

To generate the documentation for the library, run:

$ composer run docs:generate

The guide documentation pages can be found in the /guides/ directory.