# Contributing

This project is released with a [Contributor Code of Conduct][coc-md]. By participating in

this project you agree to abide by its terms.

## Request Feature

If you believe a feature is missing, report it by [creating a new issue][issue-make] or

implement it by [submitting a pull request][pr-make].

\*If you \_\_request\_\_ a new feature\*, describe precisely what you would like implemented and

be sure to completely and accurately fill out the new issue template by providing the

\*branch you would like the feature implemented on\*, whether the feature \*breaks backwards

compatibility\*, and \*any other requested details\*. Moreover, including a detailed

description, links to external resources, and other references will provide guidance and

context for whomever ultimately implementations your request.

\*If you \_\_submit\_\_ a new feature\*, ensure you provide a verbose description of your

contribution, again making sure to completely fill out the new pull request template. Your

pull request must pass our [Travis automated build check][travis], which runs our unit test

suite against various PHP versions and checks that proper code style standards are met.

## Report Bug

If you believe you have encountered a \*behavioral bug\*, \*documentation error\*, or \*other

issue\*, first search for an [existing issue][issue-list], and if one doesn't already exist,

then proceed to [create a new issue][issue-make]. If you have the available time and the

required skill set, we appreciate those who offer fixes themselves via

[pull requests][pr-list], as well.

Be sure to complete the new issue template by providing the \*version of the bundle used\*,

the \*version of Symfony used\*, and \*any other requested information\*. It is also helpful

to create a Symfony-standard fork reproducing the unexpected behavior; this provides those

trying to assist you with a working example of the bug. The more detail you provide the

more likely the issue will meet a beneficial resolution.

## Submit Your Changes

Whether you are creating a \_\_bug fix\_\_ or a \_\_new feature\_\_, the workflow for forking the

bundle, working on local changes, and finally requesting that your changes are merged back

into the upstream repository is the same.

1. \_\_Fork/Clone/Branch Repository:\_\_

Fork the repository using the [GitHub][gh] web interface, clone your fork, and create a

local `feature-` branch for new features or `bugfix-` branch for bug fixes.

```bash

# after forking, clone your fork (replace <user> with your github username)

git clone git@github.com:<user>/LiipImagineBundle.git

# enter the newly cloned repository directory

cd LiipImagineBunde

# EITHER checkout a "feature-<name>" branch if creating a new feature

git checkout -b feature-name-of-addition

# OR checkout a "bugfix-<name>" branch if working on a bug fix

git checkout -b bugfix-name-of-issue

```

2. \_\_Install Dependencies:\_\_

Install the project's dependencies using [Composer][composer]. You \*may\* also install any

of the optional dependencies defined in this project's [composer.json file][composer-file],

such as the [mongo php adapter][mongo-php] or the [monolog library][monolog], depending on

the features required.

```bash

# install required, default dependencies

composer install

# install any optional dependencies, if required

composer require namespace/package

```

\*Note: For instructions on installing [Composer][composer], reference their

official [download instructions][composer-dl].\*

3. \_\_Work/Stage/Commit (Repeat):\_\_

Make the code changes required to fix your bug or implement your new feature, then stage

your files with `git add` and commit your work with `git commit`. Your commit messages

should summarize the changes such that others can gain a general understanding of both

your literal changes and general intentions.

```bash

# stage your changed files or directories

git add foo/bar/file-a.ext

# commit your changes with a short descriptive message

git commit -m 'changed a, b, and c to implement d'

```

\*Note: It is highly recommended that you create frequent, small commits instead of large,

unorganized ones. While smaller commits are easier to review and track for the project,

they also allow you to easily revert local work if your progress detours down unproductive

path.\*

4. \_\_Add/Edit Unit Tests:\_\_

You \*must\* add tests, as appropriate for the changes and/or additions you've made to the

project. This project's test suite uses [PHPUnit][php-unit] and the unit tests can be

found in the `Tests/` directory.

```bash

# run the test suite by calling simple-phpunit

vendor/bin/simple-phpunit

```

5. \_\_Add/Edit Documentation:\_\_

All user-facing changes must be properly reflected in our documentation, whether your

changes amend or add functionality. Documentation is formatted with

[reStructuredText (RST)][rst-info] and can be found in the `Resources/doc/` directory.

These [RST][rst-info] files are automatically compiled to generate our

[official documentation website][docs].

\*Note: To test your local documentation changes, reference the

[Building RST Documentation][rst-make] wiki page where you will find detailed

information on compiling the documentation on your local environment.\*

6. \_\_Apply Code Style Standards:\_\_

You \*must\* ensure your changes abide by our code style standards before submitting any

changes. To facilitate this, we define a [code style configuration file][cs-conf] for

[php-cs-fixer][cs] that enabled you to apply our require code style in an automated

manner.

```bash

# call php-cs-fixer with "--dry-run" option to view changes without applying them

vendor/bin/php-cs-fixer fix -vvv --diff --dry-run

# to apply the code style changes, run without the "--dry-run" option

vendor/bin/php-cs-fixer fix -vvv --diff

# commit any changes made by the tool

git commit -a -m 'php-cs-fixer run'

```

\*Note: Our [automated Travis builds][travis] will mark a [pull request][pr-make] failed if

it does not meet the required code style standards.\*

7. \_\_Push to Fork/Merge Upstream:\_\_

Having completed your local changes, added tests and documentation, and ensured code

style standards are met, push your local changes back to your personal [GitHub][gh] fork

of the repository.

```bash

# push your local changes to your remote fork

git push master -u origin

```

Once you've pushed your changes to your fork, use the [GitHub][gh] web interface to

[submit a pull request][gh-help-pr].

## Additional information

\* [General GitHub documentation][gh-help]

\* [GitHub pull request documentation][gh-help-pr]

[cs-conf]: ../.php\_cs.dist

[cs]: http://cs.sensiolabs.org/

[gh-help]: https://help.github.com

[gh-help-pr]: https://help.github.com/send-pull-requests

[gh]: https://github.com

[pr-list]: https://github.com/liip/LiipImagineBundle/pulls

[pr-make]: https://github.com/liip/LiipImagineBundle/pull/new

[issue-list]: https://github.com/liip/LiipImagineBundle/issues

[issue-make]: https://github.com/liip/LiipImagineBundle/issues/new

[travis]: https://travis-ci.org/liip/LiipImagineBundle

[rst-info]: http://symfony.com/doc/current/contributing/documentation/format.html#restructuredtext

[rst-make]: https://github.com/liip/LiipImagineBundle/wiki/Building-RST-Documentation

[php-unit]: http://phpunit.de/

[composer-file]: ../composer.json

[composer]: https://getcomposer.org/

[composer-dl]: https://getcomposer.org/download/

[coc-14]: https://www.contributor-covenant.org/version/1/4/code-of-conduct.html

[coc-md]: CODE\_OF\_CONDUCT.md

[docs]: https://symfony.com/doc/master/bundles/LiipImagineBundle/index.html

[mongo-php]: https://github.com/alcaeus/mongo-php-adapter

[monolog]: https://github.com/Seldaek/monolog