Contributing

============

Before proposing a pull request, please check the following:

\* Your code should follow the [PSR-2 coding standard](https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-2-coding-style-guide.md). Use [php-cs-fixer](https://github.com/FriendsOfPHP/PHP-CS-Fixer) to fix inconsistencies.

\* If you commit a new feature, be prepared to help maintaining it. Watch the project on GitHub, and please comment on issues or PRs regarding the feature you contributed.

\* You should test your feature well.

\* You should run `php build` and check the PHAR file works before submitting your pull request. You may need to change `phar.readonly` php.ini setting to `0` or run the command as `php -d phar.readonly=0 build`.

Once your code is merged, it is available for free to everybody under the MIT License. Publishing your pull request on the PHPloy GitHub repository means that you agree with this license for your contribution.

Thank you for your contribution! PHPloy wouldn't be so great without you.

## Dependencies

To preserve compatibility with PHP 5.5 when you add new dependencies, you should try to update your composer.lock file with the `--prefer-lowest` option.

```bash

composer update --prefer-lowest

```

The downside is that you to have to set nearly exact versions for your dependencies.

## Testing

\*Special thanks to [@mbrugger](https://github.com/mbrugger) for implementing the testing functionality, a long awaited feature.\*

The integration tests provide a docker configuration for a SFTP and FTP server you can use to test the synchronization.

A folder in /tmp is mounted into the docker container which can be used to compare the result after running PHPloy.

### Executing tests

To get started with testing, please follow the steps below:

1. Install [docker](https://docs.docker.com/engine/installation/)

2. Start the test server

```

vagrant@vagrant-ubuntu-trusty-64:/vagrant/PHPloy$ ./tests/start\_test\_server.sh

```

3. run the tests

```

vagrant@vagrant-ubuntu-trusty-64:/vagrant/PHPloy$ vendor/bin/phpunit tests

PHPUnit 4.8.26 by Sebastian Bergmann and contributors.

...

Time: 2.32 seconds, Memory: 4.25MB

OK (3 tests, 4 assertions)

vagrant@vagrant-ubuntu-trusty-64:/vagrant/PHPloy$

```

4. Stop the sftp server

```

vagrant@vagrant-ubuntu-trusty-64:/vagrant/PHPloy$ ./tests/stop\_test\_server.sh

```

### Writing new tests

Basically each test should be structured in the following way

1. Prepare a test repository and phploy.ini configuration

2. Run PHPLoy

3. Verify the result of the execution by comparing the source folder with the synchronization result

```

public function testSyncAddedFileShouldSucceed($testHelper)

{

$this->testHelper = new PHPLoyTestHelper($testHelper);

$this->testHelper->givenRepositoryWithConfiguration();

$this->whenFileIsAdded();

$this->thenRepositoryIsSynchronizedSuccessfully();

}

```

### Parametrized testing SFTP and FTP

For testing FTP and SFTP at the same time tests can be parametrized. Take a look at the CommitFileTest.php:

```

class CommitFileTest extends PHPUnit\_Framework\_TestCase

{

public function provider()

{

return array(

array('ftp'),

array('sftp')

);

}

protected $testHelper;

/\*\*

\* @dataProvider provider

\*/

public function testSyncAddedFileShouldSucceed($testType)

{

$this->testHelper = new PHPLoyTestHelper($testHelper);

// implement the test here

}

}

```

The PHPLoyTestHelper is parametrized to provide different configurations based on the type passed to the test method by the PHPunit framework.

This is especially helpful if you want to test shared functionality available for multiple protocols.

### Travis CI

A [`.travis.yml`](.travis.yml) file is present on the repositoriy. You can configure your Github account and forked

branch to run this file using the [travis-ci.org](https://travis-ci.org) platform.