PHP Test Generator Manual

Release latest

Niko Granö

Feb 13, 2019

Contents

Edition for PHP Test Generator (PTG) latest. Updated on Feb 13, 2019.

PHP Test Generator (PTG) is a core component to create test cases based on project files. PHP Test Generator (PTG) comes with templates for PHPUnit by default and supports latest PHPUnit versions fully. Mocking is made with Mockery. Creating new templates are made easy and you can extend it easily at any times.

PHP Test Generator is referred as PTG later on the docs.

By Niko Granö

Contents:

CHAPTER 1

Goals

PHP Test Generator was developed with the following goals in mind:

- Easily maintainable code base.
- Be independent and modular as possible.
- Have simple install process (composer).
- Have easily extendable templates.
- Integrate easily with other than PHPUnit testing frameworks.
- Have an easy to use command-line operation.
- Have external configuration file.
- Generate stubs to speed up development, not perfect tests.

CHAPTER 2

Installation

2.1 Requirements

Note: PTG requires at least PHP 7.1 (or later).

PTG latest requires PHP 7.1; using the latest version of PHP is highly recommended. PTG has been confirmed to be working on latest PHP 7.3.

PTG requires the json and tokenizer extensions, which are normally enabled by default. PTG needs also mbstring, but this is not mandatory.

PTG also requires the pcre, reflection, and spl extensions. These standard extensions are enabled by default and cannot be disabled without patching PHP's build system and/or C sources.

2.2 Composer

Simply add a (development-time) dependency on `niko9911/php-test-generator` to your project's `composer.json` file if you use Composer to manage the dependencies of your project:

```
composer require --dev niko9911/php-test-generator ^|version|
```

2.3 Optional packages

Currently there is no optional packages available. Optional packages are supposed to provide additional features such as templates for different kind of templates or support for other testing frameworks or usages than PHPUnit.

CHAPTER 3

Commands

Currently following commands are available.

Command	Short Description
create-tests	Will create Unit Tests stubs based on configuration.

3.1 Create Tests Command

Will create Unit Tests stubs based on configuration. Configuration is automatically detected or manually given path to it by using optional parameters.

Example usage:

```
testGen create-tests
```

Available Options:

Option	Short Description
-c, –config=CONFIG	Path to configuration. Defaults to current working directory and tries to find first .php_testgen, .php_testgen.dist in given order.

3.2 Global Commands and Options

As tool is based on Symfony Console you can use default commands to get more information like `testGen -h` or `testGen [command] -h`

CHAPTER 4

Configuration

PTG can be only configured via using external configuration file. PTG will try find configuration file in following order:

- Read Option parameter `--config path/to/config.php` (`-c ...`).
- Try find `.php_testgen` in current working directory.
- try find `.php_testgen.dist` in current working directory.

If `.php_testgen` and `.php_testgen.dist` exist at same time `.php_testgen` will be used as source for the configuration. This way you can override configuration locally if needed.

4.1 Configuration File

Configuration in PTG works by creating PHP file which will return instance on ConfigurationBuilder. Any other return than ConfigurationBuilder will result ConfigurationError and script will not run.

Example quick configuration:

```
<?php

declare(strict_types=1);

use Lamia\\TestGenerator\Domain\Builder\ConfigurationBuilder as Config;
use Lamia\\TestGenerator\Domain\Configuration\DirectoryConfigurationEntity as␣
↪Directory;

return (new Config())
    ->setTemplate(Config::TEMPLATE_PHPUNIT)
    ->addDirectory(new Directory('src', 'tests/src/unit', [], [], ['integration',
↪'config', 'etc'], true))
    ->addDirectory(new Directory('lib', 'tests/lib/unit', [], [], ['stubs'], false));
```

Directory Object Explained:

```
new Directory(
    string sourceDirectory,
    string testDirectory,
    Lamia\\TestGenerator\Domain\Tag[] Tags,
    Lamia\\TestGenerator\Domain\Option[] Options,
    string[] ExcludedDirectories,
    bool AutoTag
  );
```

4.2 Available Global Tags

For full list of available tags, please consult correct template section. Global Tags available are following:

Tag	Usage	Description
Lamia\TestGenerator\Domain\Tag\Custom::class	new Custom(string $name, string $value);	Adds custom tag into every function doc.

4.3 Available Global Options

For full list of available options, please consult correct template section. Global Options available are following:

Options	Usage	Description
None	None	None

4.4 Available Templates

Here is available templates for now. If you have template to submit, please create PR to add it to the list.

Template	Alias	Description
\Lamia\TestGenerator\Template\PhpUnit\Template::class	Config::TEMPLATE_PHPUNIT	Default Template for PHPUnit.

*) Requires extra Composer Package to be installed.

CHAPTER 5

PHPUnit

This is template for PHPUnit. It will create basic structure to get started with PHPUnit. Some tests generated with this template might work without any problems, but most of the automatically generated code will not work and will need manual fixes.

5.1 Available Tags

These tags can be used only if the template is PHPUnit. PHPUnit template will provide following tags:

5.2 Available Options

These options can be used only if the template is PHPUnit. PHPUnit template will provide following options:

Options	Usage	Description
Lamia\TestGenerator\Template\PhpUnit\Domain\Option\CustomExtend	new CustomExtend($class);	Custom class to extend. Must be TestCase at last.
Lamia\TestGenerator\Template\PhpUnit\Domain\Option\CustomFaker	new CustomFaker($faker);	Custom command to call faker. Must return \Faker\Generator:class
Lamia\TestGenerator\Template\PhpUnit\Domain\Option\NamespaceGroups	new NamespaceGroups($enabled = true);	Add @groups for every namespace part.
Lamia\TestGenerator\Template\PhpUnit\Domain\Option\UseCoversSelf	new UseCoversSelf($enabled = true);	Adds @covers tag to the function to prevent false positive coverage.
Lamia\TestGenerator\Template\PhpUnit\Domain\Option\UseCoversDefaultClass	new UseCoversDefaultClass($enabled = true);	Adds @defaultCovers to enable shorthands for the class.

5.3 Configuration

Example quick configuration based on default configuration:

```php
<?php

declare(strict_types=1);

use Lamia\TestGenerator\Domain\Builder\ConfigurationBuilder as Config;
use Lamia\TestGenerator\Domain\Configuration\DirectoryConfigurationEntity as␣
↪Directory;

$srcTags = [];
$srcTags[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Tag\Author('Niko');
$srcTags[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Tag\Ticket('SMT-593');

$libTags = [];
$libTags[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Tag\Group('Integration
↪');
$libTags[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Tag\Ticket('INT-31');

$srcOpt = [];
$srcOpt[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Option\CustomFaker('
↪$this->getFaker()');
$srcOpt[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Option\UseCovers(true);
$srcOpt[] = new␣
↪\Lamia\TestGenerator\Template\PhpUnit\Domain\Option\UseCoversDefault(true);
$srcOpt[] = new␣
↪\Lamia\TestGenerator\Template\PhpUnit\Domain\Option\NamespaceGrouper(true);


$libOpt = [];
$libOpt[] = new␣
↪\Lamia\TestGenerator\Template\PhpUnit\Domain\Option\CustomExtend(\Vendor\Project\Tests\AbstractUnit
↪
$libOpt[] = new \Lamia\TestGenerator\Template\PhpUnit\Domain\Option\UseCovers(true);
$libOpt[] = new␣
↪\Lamia\TestGenerator\Template\PhpUnit\Domain\Option\UseCoversDefault(true);

return (new Config())
    ->setTemplate(Config::TEMPLATE_PHPUNIT)
    ->addDirectory(new Directory('src', 'tests/src/unit', $srcTags, $srcOpt, ['config
↪', 'etc'], true))
    ->addDirectory(new Directory('lib', 'tests/lib/unit', $libTags, $libOpt, ['stubs
↪'], false));
```

5.4 Example Output

```php
<?php

declare(strict_types=1);

namespace Vendor\Project\Tests\Unit\User;

use Vendor\Project\User\UserId;
use Vendor\Project\User\SupportNumber;
use Vendor\Project\Exception\User\Account\CountryException;
use Vendor\Project\Tests\AbstractUnitTest;
```

```php
final class UserTest extends AbstractUnitTest
{
    /** @var UserId */
    private $id;

    /** @var SupportNumber */
    private $SupportNumber;

    /** @var string */
    private $country;

    /** @var DateTimeImmutable */
    private $updated;

    public function setUp(): void
    {
        $this->id = \Mockery::mock(UserId::class);
        $this->SupportNumber = \Mockery::mock(SupportNumber::class);
        $this->country = $this->getFaker()->word;
        $this->updated = \Mockery::mock(\DateTimeImmutable::class);
    }


    /**
     * @testdox Will test class Account function __construct.
     * @throws CountryException
     * @return Account
     * @group Vendor
     * @group Project
     * @group User
     * @author Niko
     * @ticket SMT-593
     * @covers Account::__construct
     * @small
     */
    public function test__construct(): Account
    {
        $class = new Account(
            $this->id,
            $this->SupportNumber,
            $this->country,
            $this->updated
        );
        $this->addToAssertionCount(1);

        return $class;
    }

    /**
     * @testdox Will test class Account function __construct for exception␣
↪CountryException.
     * @throws CountryException
     * @return void
     * @group Vendor
     * @group Project
     * @group User
     * @author Niko
     * @ticket SMT-593
```

```
     * @covers Account::__construct
     * @expectedException CountryException
     * @small
     */
    public function test__constructCountryException(): void
    {
        $this->markTestSkipped('TODO: Implement Test::__constructCountryException().
↪');
        // TODO: Please implement this manually to trigger given exception!

        new Account();
    }

    /**
     * @testdox Will test class Account function id.
     * @param Account $class
     * @return void
     * @group Vendor
     * @group Project
     * @group User
     * @author Niko
     * @ticket SMT-593
     * @covers Account::id
     * @depends test__construct
     * @small
     */
    public function testId(Account $class): void
    {
        $this->assertEquals($this->id, $class->id());
    }

    /**
     * @testdox Will test class Account function SupportNumber.
     * @param Account $class
     * @return void
     * @group Vendor
     * @group Project
     * @group User
     * @author Niko
     * @ticket SMT-593
     * @covers Account::supportNumber
     * @depends test__construct
     * @small
     */
    public function testSupportNumber(Account $class): void
    {
        $this->assertEquals($this->SupportNumber, $class->supportNumber());
    }

    /**
     * @testdox Will test class Account function country.
     * @param Account $class
     * @return void
     * @group Vendor
     * @group Project
     * @group User
     * @author Niko
     * @ticket SMT-593
```

```
     * @covers Account::country
     * @depends test__construct
     * @small
     */
    public function testCountry(Account $class): void
    {
        $this->assertEquals($this->country, $class->country());
    }

    /**
     * @testdox Will test class Account function updated.
     * @param Account $class
     * @return void
     * @group Vendor
     * @group Project
     * @group User
     * @author Niko
     * @ticket SMT-593
     * @covers Account::updated
     * @depends test__construct
     * @small
     */
    public function testUpdated(Account $class): void
    {
        $this->assertEquals($this->updated, $class->updated());
    }
}
```

CHAPTER 6

Contributing to PHP Test Generator

6.1 Contributor Code of Conduct

Please note that this project is released with a :ref‘code_of_conduct‘. By participating in this project you agree to abide by its terms.

6.2 Workflow

- Fork the project.
- Make your bug fix or feature addition.
- Add tests for it. This is important so we don’t break it in a future version unintentionally.
- Send a pull request. Bonus points for topic branches.

Please make sure that you have set up your user name and email address for use with Git. Strings such as `silly nick name <root@localhost>` look really stupid in the commit history of a project.

Pull requests for bug fixes must be based on the current stable branch whereas pull requests for new features must be based on the `master` branch.

We are trying to keep backwards compatibility breaks in PHP Test Generator to an absolute 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.

6.3 Coding Guidelines

This project comes with a configuration file and an executable for php-cs-fixer (`.php_cs.dist`) that you can use to (re)format your source code for compliance with this project’s coding guidelines:

```
vendor/bin/php-cs-fixer fix
```

6.4 Using PHP Test Generator from a Git checkout

The following commands can be used to perform the initial checkout of PHP Test Generator:

```
git clone https://gitlab.com/niko9911/php-test-generator.git ptg

cd ptg
```

Retrieve PHP Test Generator's dependencies using Composer:

```
composer install
```

The `testGen` script can be used to invoke the PHP Test Generator test runner:

```
./testGen --version
```

6.5 Running PHP Test Generator's test suite

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

```
vendor/bin/phpunit
```

6.6 Reporting issues

Please use report all tickets to:

- General problems/Documentation

CHAPTER 7

Contributor Code of Conduct

As contributors and maintainers of this project, and in the interest of fostering an open and welcoming community, we pledge to respect all peoplewho contribute through reporting issues, posting feature requests, updating documentation, submitting pull requests or patches, and other activities.

We are committed to making participation in this project a harassment-free experience for everyone, regardless of level of experience, gender, gender identity and expression, sexual orientation, disability, personal appearance, body size, race, ethnicity, age, religion, or nationality.

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery
- Personal attacks
- Trolling or insulting/derogatory comments
- Public or private harassment
- Publishing other's private information, such as physical or electronic addresses, without explicit permission
- Other unethical or unprofessional conduct

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

By adopting this Code of Conduct, project maintainers commit themselves to fairly and consistently applying these principles to every aspect of managing this project. Project maintainers who do not follow or enforce the Code of Conduct may be permanently removed from the project team.

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community.

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project maintainer at niko9911@ironlions.fi. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. Maintainers are obligated to maintain confidentiality with regard to the reporter of an incident.

This Code of Conduct is adapted from the Contributor Covenant, version 1.3.0, available at https://contributor-covenant.org/version/1/3/0/

CHAPTER 8

Copyright

```
                        Copyright (c) 2019 Niko Granö.

                    This work is licensed under the MIT License.

    A summary of the license is given below, followed by the full legal text.

                        ! Summary is not a license !
                Full legal text is always legally used in case of
                  conflicts between summary and license itself.

-------------------------------------------------------------------------------

        Basically, you can do whatever you want as long as you include the
    original copyright and license notice in any copy of the software/source.

          *---------------------------------------------------------*
          |        CAN          |     CANNOT    |        MUST       |
          ----------------------+---------------+--------------------
          |  COMMERCIAL USE     | HOLD LIABLE   | INCLUDE COPYRIGHT   |
          |  MODIFY             |               | INCLUDE LICENSE     |
          |  DISTRIBUTE         |               |                     |
          |  SUBLICENSE         |               |                     |
          |  PRIVATE USE        |               |                     |
          *---------------------------------------------------------*

-------------------------------------------------------------------------------

   MIT License

   Copyright (c) 2019 Niko Granö

   Permission is hereby granted, free of charge, to any person obtaining a copy
   of this software and associated documentation files (the "Software"), to deal
   in the Software without restriction, including without limitation the rights
   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
```

```
   copies of the Software, and to permit persons to whom the Software is
   furnished to do so, subject to the following conditions:

   The above copyright notice and this permission notice shall be included in all
   copies or substantial portions of the Software.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
   SOFTWARE.



================================================================================
```