**PHPUnit —Аннотации**

**[@author](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.author)**

**[@backupGlobals](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.backupGlobals)**

**[@backupStaticAttributes](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.backupStaticAttributes)**

**[@codeCoverageIgnore\*](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.codeCoverageIgnore)**

**[@covers](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.covers)**

**[@coversNothing](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.coversNothing)**

**[@dataProvider](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.dataProvider)**

**[@depends](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.depends)**

**[@expectedException](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.expectedException)**

**[@expectedExceptionCode](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.expectedExceptionCode)**

**[@expectedExceptionMessage](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.expectedExceptionMessage)**

**[@group](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.group)**

**[@outputBuffering](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.outputBuffering)**

**[@requires](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.requires)**

**[@runTestsInSeparateProcesses](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.runTestsInSeparateProcesses)**

**[@runInSeparateProcess](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.runInSeparateProcess)**

**[@test](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html" \l "appendixes.annotations.test)**

[**@testdox**](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#appendixes.annotations.testdox)

[**@ticket**](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#appendixes.annotations.ticket)

Аннотации - это специальная форма метаданных, которая может быть добавлена в исходный код некоторых языков программирования. Так как PHP не имеет возможности аннотирования исходного кода, PHP сообщество приняло использование тегов **@annotation arguments** в docblock для аннотаций. В PHP блоки документации доступны с помощью метода **getDocComment()** из Reflection API, применяемого к классу, методу или атрибуту. Приложения, такие как PHPUnit, используют эту информацию чтобы изменять свое поведение в процессе выполнения.

Это приложение содержит все возможные аннотации, поддерживаемые PHPUnit.

**@author**

Аннотация **@author** является синонимом аннотации **@group** (см. [«**@group**](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#appendixes.annotations.group)») и позволяет фильтровать тесты по их авторам.

**@backupGlobals**

Операции сохранения и восстановления глобальных переменных можно полностью выключить для тест-кейса с помощью следующего:

/\*\*

\* @backupGlobals disabled

\*/

class MyTest extends PHPUnit\_Framework\_TestCase

{

// ...

}

Аннотацию **@backupGlobals** так же можно использовать на уровне методов тест-кейса. Это позволяет тонко настроить операции сохранения и восстановления:

/\*\*

\* @backupGlobals disabled

\*/

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @backupGlobals enabled

\*/

public function testThatInteractsWithGlobalVariables()

{

// ...

}

}

**@backupStaticAttributes**

Операции сохранения и восстановления статических атрибутов можно выключить для всего класса теста следующим образом:

/\*\*

\* @backupStaticAttributes disabled

\*/

class MyTest extends PHPUnit\_Framework\_TestCase

{

// ...

}

Аннотация **@backupStaticAttributes** может быть использована на уровне методов. Это позволяет тонко настроить операции сохранения и восстановления:

/\*\*

\* @backupStaticAttributes disabled

\*/

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @backupStaticAttributes enabled

\*/

public function testThatInteractsWithStaticAttributes()

{

// ...

}

}

**@codeCoverageIgnore\***

Аннотации **@codeCoverageIgnore**, **@codeCoverageIgnoreStart** и **@codeCoverageIgnoreEnd** можно использовать для исключения участков кода из статистики покрытия кода.

Примеры использования можно посмотреть в [«Ignoring Code Blocks»](https://phpunit.de/manual/current/en/code-coverage-analysis.html#code-coverage-analysis.ignoring-code-blocks).

**@covers**

Аннотация **@covers** используется для указания того, какие метод(ы) этот тест-метод будет тестировать:

/\*\*

\* @covers BankAccount::getBalance

\*/

public function testBalanceIsInitiallyZero()

{

$this->assertEquals(0, $this->ba->getBalance());

}

Если указана эта аннотация, то в статистике покрытия кода этого тест-метода будут учитываться только указанные в аннотации метод(ы).

[Таблица B.1, «Аннотации для указания того, какие методы покрывает тест»](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#appendixes.annotations.covers.tables.annotations) показывает синтаксис аннотации **@covers**.

**Таблица B.1. Аннотации для указания того, какие методы покрывает тест**

**Аннотация**

**Описание**

**@covers ClassName::methodName**

**Указывает что метод теста покрывает указанный метод.**

**@covers ClassName**

**Указывает что метод теста покрывает все методы указанного класса.**

**@covers ClassName<extended>**

**Указывает что метод теста покрывает все методы указанного класса, включая родительские класс(ы) и интерфейс(ы).**

**@covers ClassName::<public>**

**Указывает что тест метод покрывает все public методы указанного класса.**

**@covers ClassName::<protected>**

**Указывает что тест метод покрывает все protected методы указанного класса.**

**@covers ClassName::<private>**

**Указывает что тест метод покрывает все private методы указанного класса.**

**@covers ClassName::<!public>**

**Указывает что тест метод покрывает все методы указанного класса, кроме public**

**@covers ClassName::<!protected>**

**Указывает что тест метод покрывает все методы указанного класса, кроме protected.**

**@covers ClassName::<!private>**

**Указывает что тест метод покрывает все методы указанного класса, кроме private.**

**@coversNothing**

Аннотация **@coversNothing** используется в коде теста для указания того, что информация о покрытии кода аннотированным классом не будет записана.

Эту аннотацию можно использовать для интеграционных тестов. См. [Пример 14.3, «A test that specifies that no method should be covered»](https://phpunit.de/manual/current/en/code-coverage-analysis.html" \l "code-coverage-analysis.specifying-covered-methods.examples.GuestbookIntegrationTest.php" \o "Пример 14.3. A test that specifies that no method should be covered)

Аннотацию можно использовать на уровне класса и метода. Она заменит все другие аннотации **@covers**.

**@dataProvider**

Метод теста может принимать агрументы. Эти агрументы предоставляет метод провайдера данных (data provider) (**provider()** описан в [Пример 4.4, «Использование провайдера данных, который возвращает массив массивов »](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.data-providers.examples.DataTest.php)). Метод поставщика данных указывается с помощью аннотации **@dataProvider** .

Более детальная информация [«Провайдеры данных»](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.data-providers).

**@depends**

PHPUnit поддерживает указание зависимостей между методами тестов. Зависимости не указывают порядок выполнения методов теста, но они позволяют передавать объект с тестовыми данными (фикстурами), созданный поставщиком (producer) к потребителям (consumers), котрые от него зависят. [Пример 4.2, «Использвание аннотации](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.examples.StackTest2.php) [**@depends**](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.examples.StackTest2.php) для описания зависимостей » показывает пример использования аннотации **@depends** для указания зависимостей между методами теста.

Для более детальной информации см. [«Межтестовые зависимости»](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.test-dependencies)

**@expectedException**

[Пример 4.7, «Использование аннотации @expectedException»](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.exceptions.examples.ExceptionTest.php) показывает пример использования аннотации **@expectedException** для тестирования исключений (exceptions) в тестируемом коде.

Для более детальной информации см. [«Тестирование исключений»](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#writing-tests-for-phpunit.exceptions)

**@expectedExceptionCode**

Аннотация **@expectedExceptionCode** совместно с аннотацией **@expectedException** позволяет указать какой именно код исключения следует ожидать.

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @expectedException MyException

\* @expectedExceptionCode 20

\*/

public function testExceptionHasErrorcode20()

{

throw new MyException('Some Message', 20);

}

}

Для облегчения тестирования и уменьшения копирования кода можно использовать короткий вариант записи с использованием константы в **@expectedExceptionCode** используя синтаксис: "**@expectedExceptionCode ClassName::CONST**".

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @expectedException MyException

\* @expectedExceptionCode MyClass::ERRORCODE

\*/

public function testExceptionHasErrorcode20()

{

throw new MyException('Some Message', 20);

}

}

class MyClass

{

const ERRORCODE = 20;

}

**@expectedExceptionMessage**

Аннотация **@expectedExceptionMessage** работает по схожему принципу с **@expectedExceptionCode** и позволяет сделать утверждение (assertion) о сообщении в исключении.

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @expectedException MyException

\* @expectedExceptionMessage Some Message

\*/

public function testExceptionHasRightMessage()

{

throw new MyException('Some Message', 20);

}

}

Ожидаемое сообщение может быть подстрокой сообщения исключения. Может быть полезным указывать определённое имя или параметр в исключении, чтобы не фиксировать полное исключение в коде теста.

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @expectedException MyException

\* @expectedExceptionMessage broken

\*/

public function testExceptionHasRightMessage()

{

$param = "broken";

throw new MyException('Invalid parameter "'.$param.'".', 20);

}

}

Для облегчения тестирования и уменьшения копирования кода можно использовать короткий вариант записи с использованием константы в **@expectedExceptionMessage** используя синтаксис: "**@expectedExceptionMessage ClassName::CONST**". Пример можно посмотреть в [«**@expectedExceptionCode**](http://yapro.ru/web-master/php/writing-tests-for-phpunit.html#appendixes.annotations.expectedExceptionCode)».

**@group**

Тест можно присвоить какой-либо группе тестов. Можно указывать одну или несколько групп одновременно. Пример использования аннотации **@group** :

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @group specification

\*/

public function testSomething()

{

}

/\*\*

\* @group regresssion

\* @group bug2204

\*/

public function testSomethingElse()

{

}

}

Выполняемые группы тестов можно выбрать используя параметры **--group** и **--exclude-group** при вызове PHPUnit из командной строки, или с помощью конфигурации XML.

**@outputBuffering**

Аннотация **@outputBuffering** используется для [контроля вывода](http://yapro.ru/system/redirect?to=http://www.php.net/manual/ru/intro.outcontrol.php) в PHP

/\*\*

\* @outputBuffering enabled

\*/

class MyTest extends PHPUnit\_Framework\_TestCase

{

// ...

}

Аннотацию **@outputBuffering** также можно использовать на уровне методов класса. Это позволяет более точно управлять выводом:

/\*\*

\* @outputBuffering disabled

\*/

class MyTest extends PHPUnit\_Framework\_TestCase

{

/\*\*

\* @outputBuffering enabled

\*/

public function testThatPrintsSomething()

{

// ...

}

}

**@requires**

Аннотация **@requires** используется для пропуска тестов, когда, например, версия PHP не соответствует используемой.

Полный список примеров описан в [Таблица 9.3, «Possible @requires usages»](https://phpunit.de/manual/current/en/incomplete-and-skipped-tests.html#incomplete-and-skipped-tests.requires.tables.api)

**@runTestsInSeparateProcesses**

**@runInSeparateProcess**

**@test**

Вместо префиксов **test** к методам, можно использовать аннотацию **@test** в DocBlock для пометки метода как теста.

/\*\*

\* @test

\*/

public function initialBalanceShouldBe0()

{

$this->assertEquals(0, $this->ba->getBalance());

}

**@testdox**

**@ticket**