

# PHPUnit e metodologia TDD

## Rimorsoft Online (Italo Morales)

[https://www.youtube.com/watch?v=1kCy\\_7Bu3Mc&list=PLhCiuvlix-rR3Pk8uaPN3duC1JHaibSrN](https://www.youtube.com/watch?v=1kCy_7Bu3Mc&list=PLhCiuvlix-rR3Pk8uaPN3duC1JHaibSrN)

Resumo do curso feito por Roberto Pinheiro

### Aula 01 - Introdução

**PHPUnit** é um framework para provar profissionalmente código em PHP. Nesta série entenderemos como funciona e também conheceremos a metodologia TDD, que significa desenvolvimento guiado por provas.

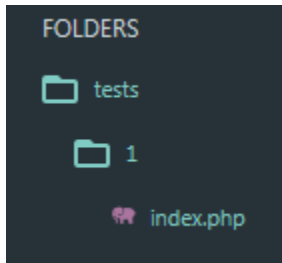
**TDD (Test Drive - Development ou desenvolvimento guiado por provas)** é uma metodologia de desenvolvimento.

Cada linguagem de programação tem sua própria ferramenta de provas.

PHPUnit é uma ferramenta (framework) de provas PHP.

## Aula 02 - Fluxo de trabalho sem PHPUnit

Crie a subpasta **1** (exercício 1)



### index.php

```
<?php
```

```
class Slug
```

```
{
```

```
    public function render($original)
```

```
    {
```

```
        $slug = str_replace(" ", "-", $original);
```

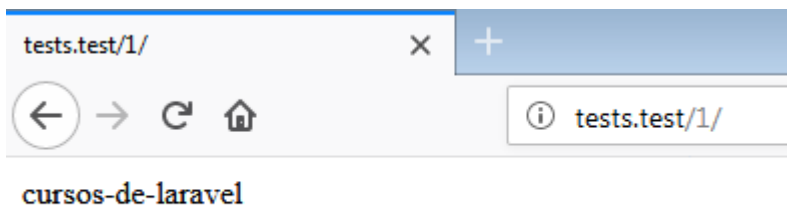
```
        return strtolower($slug);
```

```
    }
```

```
}
```

```
$slug = new Slug;
```

```
echo $slug->render("Cursos de Laravel");
```



Fazendo uma melhoria:

### index.php

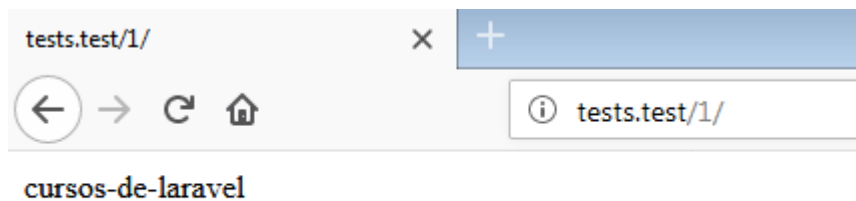
```
{
    protected $original;

    public function __construct($original)
    {
        $this->original = $original;
    }

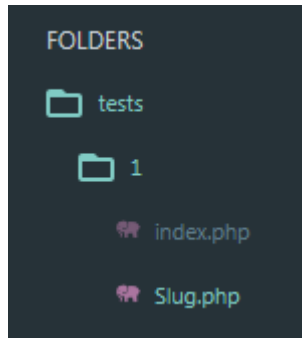
    public function render()
    {
        $slug = str_replace(" ", "-", $this->original);
        return strtolower($slug);
    }
}
```

```
$slug = new Slug("Cursos de Laravel");
```

```
echo $slug->render();
```



Fazendo uma nova melhoria:



### index.php

```
require('Slug.php');

$slug = new Slug("Cursos de Laravel");

echo $slug->render();
```

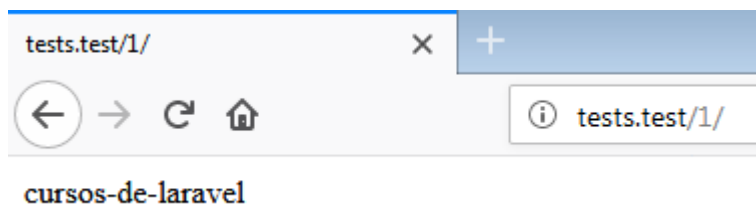
### Slug.php

```
<?php

class Slug
{
    protected $original;

    public function __construct($original)
    {
        $this->original = $original;
    }

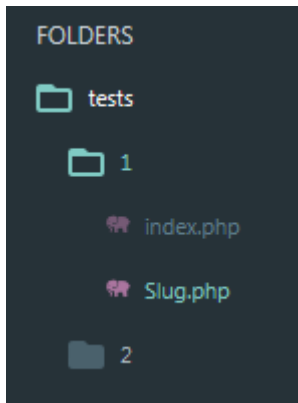
    public function render()
    {
        $slug = str_replace(" ", "-", $this->original);
        return strtolower($slug);
    }
}
```



## Aula 03 - Primeira prova com PHPUnit

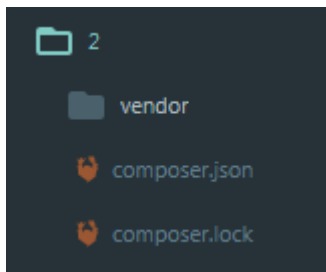
Na aula anterior ilustramos o processo de desenvolvimento sem provas, aqui buscamos recriar esse exercícios usando PHPUnit.

Dentro da pasta **tests**, crie a subpasta **2** (exercício 2).



Para instalar o PHPUnit, dentro da subpasta **2**, nessa subpasta entre com o seguinte comando:

```
composer require --dev phpunit/phpunit ^8
```



### composer.json

```
{
  "require-dev": {
    "phpunit/phpunit": "8"
  }
}
```

Dentro da subpasta 2, crie o arquivo SlugTest.php

### SlugTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class SlugTest extends TestCase
{
    public function test_render()
    {
        require('Slug.php');

        $slug = new Slug("Cursos de Laravel");

        $expected = "cursos-de-laravel";

        $this->assertEquals($slug->render(), $expected);
    }
}
```

E dentro da subpasta **2**, também crie o arquivo **Slug.php**:

### Slug.php

```
<?php

class Slug
{
    protected $original;

    public function __construct($original)
    {
        $this->original = $original;
    }

    public function render()
    {
        $slug = str_replace(" ", "-", $this->original);
        return strtolower($slug);
    }
}
```

assertion = asserção ou afirmação
-----------------------------------

Entre com o comando:

```
vendor\bin\phpunit SlugTest.php --color
```

```
C:\laragon\www\tests\2
λ vendor\bin\phpunit SlugTest.php --color
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.                                                                    1 / 1 (100%)

Time: 85 ms, Memory: 4.00 MB

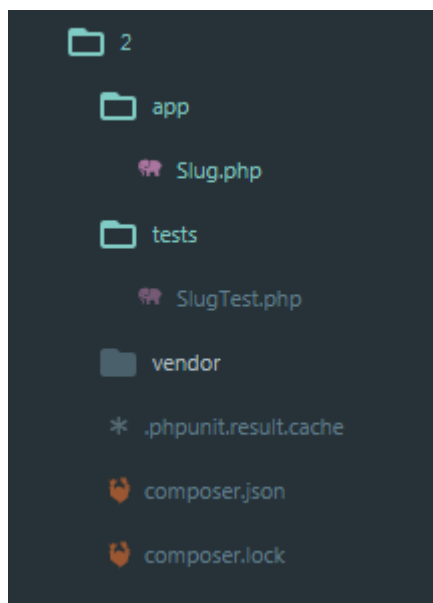
OK (1 test, 1 assertion)
C:\laragon\www\tests\2
λ |
```

Para uma melhor organização, crie dentro da subpasta **2**, duas outras subpastas:

- **app**
- **tests**

Mova o arquivo **Slug.php** para a subpasta **app**.

Mova o arquivo **SlugTest.php** para a subpasta **tests**.



Altere o arquivo **SlugTest.php** para:

## SlugTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class SlugTest extends TestCase
{
    public function test_render()
    {
        require('app\Slug.php');

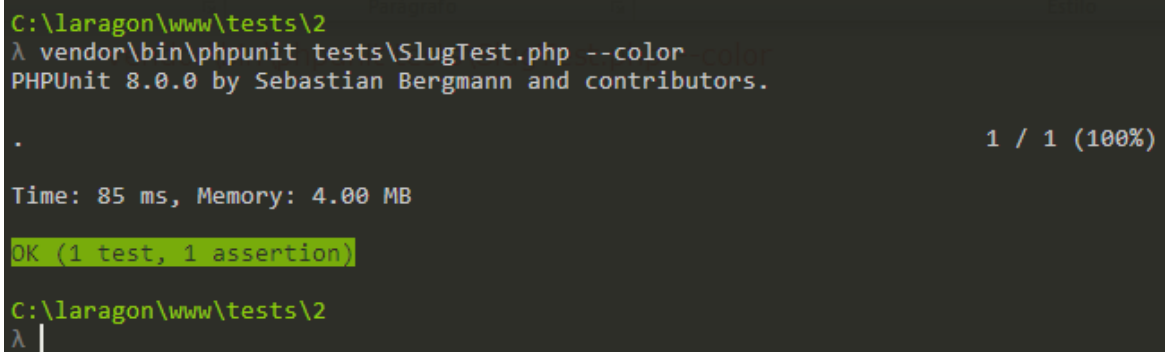
        $slug = new Slug("Cursos de Laravel");

        $expected = "cursos-de-laravel";

        $this->assertEquals($slug->render(), $expected);
    }
}
```

Entre com o comando

vendor\bin\phpunit tests\SlugTest.php --color



```
C:\laragon\www\tests\2
λ vendor\bin\phpunit tests\SlugTest.php --color
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.                                                                    1 / 1 (100%)

Time: 85 ms, Memory: 4.00 MB

OK (1 test, 1 assertion)

C:\laragon\www\tests\2
λ |
```

O resultado deve ser o mesmo.



## Aula 04 - Estrutura de pastas

Nesta aula instalaremos o Laravel para observar sua estrutura, a ideia é criar exercícios muito próximos a estrutura estudada anteriormente para que quando trabalharmos com o Laravel isso se torne bastante familiar e possamos entender melhor cada conceito.

Na subpasta **2**, suba um nível com:

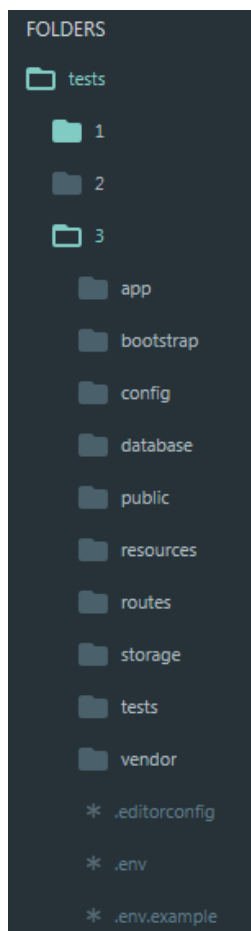
```
cd..
```

Agora entre com o comando:

```
laravel new 3
```

```
C:\laragon\www\tests
λ laravel new 3
```

Será criada a subpasta **3** com os arquivos do Laravel:



## Aula 05 - Estrutura de provas como no Laravel

### Criando o exercício 2 com a estrutura do Laravel

Crie a subpasta 4 (exercício 4). Dentro dela instalar o PHPUnit:

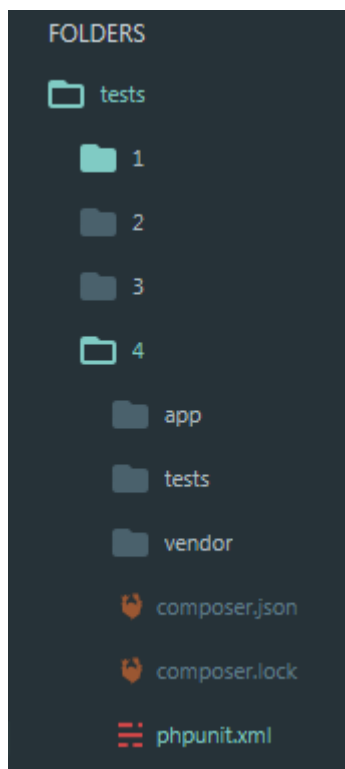
```
composer require --dev phpunit/phpunit ^8
```

```
C:\laragon\www\tests\4  
λ composer require --dev phpunit/phpunit ^8
```

Dentro da subpasta 4, crie as subpastas:

- app
- tests

E crie o arquivo `phpunit.xml`



## phpunit.xml

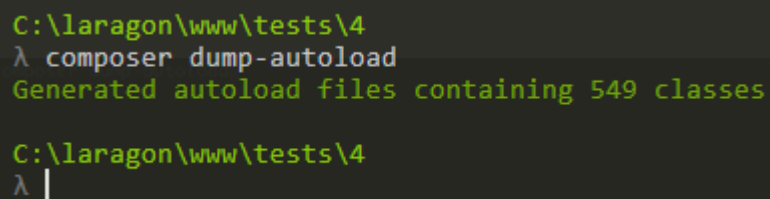
```
<?xml version="1.0" encoding="UTF-8"?>
<phpunit bootstrap="vendor/autoload.php"
  colors="true">
  <testsuites>
    <testsuite name="Application Test Suite">
      <directory suffix="Test.php">./tests</directory>
    </testsuite>
  </testsuites>
</phpunit>
```

## composer.json

```
{
  "require-dev": {
    "phpunit/phpunit": "8"
  },
  "autoload": {
    "psr-4": {
      "App\\": "app"
    }
  }
}
```

Entre com o comando:

`composer dump-autoload`



```
C:\laragon\www\tests\4
λ composer dump-autoload
Generated autoload files containing 549 classes

C:\laragon\www\tests\4
λ |
```

Dentro da subpasta `tests`, crie o arquivo `SlugTest.php`:

### SlugTest.php

```
<?php

use PHPUnit\Framework\TestCase;
use App\Slug;

class SlugTest extends TestCase
{
    public function test_render()
    {
        $slug = new Slug("Cursos de Laravel");
        $this->assertEquals($slug->render(), "cursos-de-laravel");
    }
}
```

Dentro da subpasta **tests** crie o arquivo **Slug.php**:

### Slug.php

```
<?php

namespace App;

class Slug
{
    protected $original;

    public function __construct($original)
    {
        $this->original = $original;
    }

    public function render()
    {
        $slug = str_replace(" ", "-", $this->original);
        return strtolower($slug);
    }
}
```

Entre com o comando:

vendor/bin/phpunit

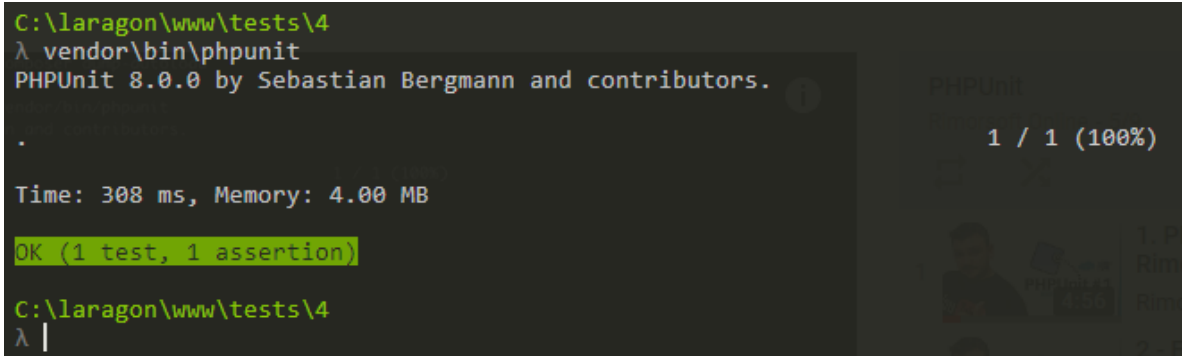
```
C:\laragon\www\tests\4
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.

Time: 308 ms, Memory: 4.00 MB

OK (1 test, 1 assertion)

C:\laragon\www\tests\4
λ |
```



## Criando um novo teste

### SlugTest.php

```
<?php
```

```
use PHPUnit\Framework\TestCase;
```

```
use App\Slug;
```

```
class SlugTest extends TestCase
```

```
{
```

```
    public function test_render()
```

```
    {
```

```
        $slug = new Slug("Cursos de Laravel");
```

```
        $this->assertEquals($slug->render(), "cursos-de-laravel");
```

```
    }
```

```
    public function test_render_without_spaces()
```

```
    {
```

```
        $slug = new Slug(" Cursos de Laravel ");
```

```
        $this->assertEquals($slug->render(), "cursos-de-laravel");
```

```
    }
```

```
}
```

Testando:

```
C:\laragon\www\tests\4
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

SlugTest.php
.F                                                                    2 / 2 (100%)

Time: 107 ms, Memory: 4.00 MB

There was 1 failure:
1) SlugTest::test_render_without_spaces
Failed asserting that two strings are equal.
--- Expected
+++ Actual
@@ @@
- '---cursos-de-laravel----'
+ 'cursos-de-laravel'

C:\laragon\www\tests\4\tests\SlugTest.php:17 >render(), "cursos-de-laravel");

FAILURES!
Tests: 2, Assertions: 2, Failures: 1.

C:\laragon\www\tests\4
λ |
```

Corrigindo:

### Slug.php

```
<?php
```

```
namespace App;
```

```
class Slug
```

```
{
```

```
    protected $original;
```

```
    public function __construct($original)
```

```
    {
```

```
        $this->original = $original;
```

```
    }
```

```
    public function render()
```

```
    {
```

```
        $slug = str_replace(" ", "-", trim($this->original));
```

```
        return strtolower($slug);
```

```
    }
```

```
}
```

Testando:

```
C:\laragon\www\tests\4
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..
Time: 176 ms, Memory: 4.00 MB

OK (2 tests, 2 assertions)

C:\laragon\www\tests\4
λ |
```

Cada método é uma prova e cada método tem uma afirmação (assertion).

## Uma correção de código

### Slug.php

```
<?php
```

```
namespace App;
```

```
class Slug
```

```
{
    protected $original;

    public function __construct($original)
    {
        $this->original = trim($original);
    }
    public function render()
    {
        $slug = str_replace(" ", "-", $this->original);
        return strtolower($slug);
    }
}
```

```
C:\laragon\www\tests\4
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..
Time: 96 ms, Memory: 4.00 MB

OK (2 tests, 2 assertions)

C:\laragon\www\tests\4
λ |
```

## Outra correção de código (melhoria)

### Slug.php

```
<?php

namespace App;

class Slug
{
    protected $original;

    public function __construct($original)
    {
        $this->original = $original;
    }

    public function getOriginal()
    {
        return trim($this->original);
    }

    public function render()
    {
        $slug = str_replace(" ", "-", $this->getOriginal());
        return strtolower($slug);
    }
}
```

```
C:\laragon\www\tests\4
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..                                                                  2 / 2 (100%)

Time: 97 ms, Memory: 4.00 MB

OK (2 tests, 2 assertions)
C:\laragon\www\tests\4
λ
```



## Aula 06 - Asserções ou afirmações em PHPUnit

Crie a subpasta **5** (exercício 5) e dentro dela instale o **PHPUnit**.

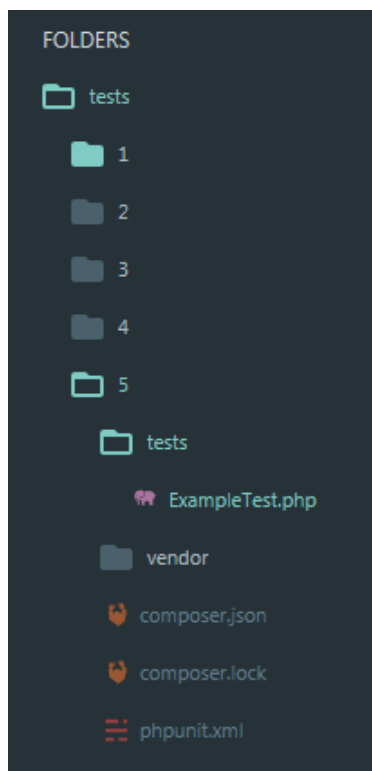
```
composer require --dev phpunit/phpunit ^8
```

Dentro da subpasta **5** crie a subpasta **tests**. E dentro dela crie o arquivo **ExampleTest.php**

Crie o arquivo **phpunit.xml**

### phpunit.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<phpunit bootstrap="vendor/autoload.php"
  colors="true">
  <testsuites>
    <testsuite name="Application Test Suite">
      <directory suffix="Test.php">./tests</directory>
    </testsuite>
  </testsuites>
</phpunit>
```

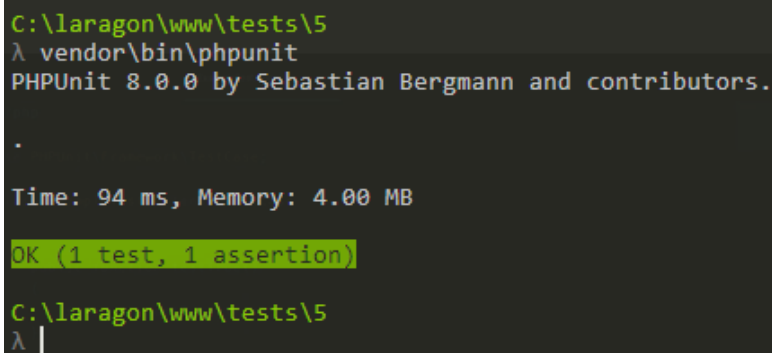


## ExampleTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }
}
```

A terminal window showing the execution of PHPUnit. The command 'vendor/bin/phpunit' is run from the directory 'C:\laragon\www\tests\5'. The output shows 'PHPUnit 8.0.0 by Sebastian Bergmann and contributors.' followed by a single dot '.' representing a passed test. Below this, it shows 'Time: 94 ms, Memory: 4.00 MB' and 'OK (1 test, 1 assertion)' in a green box. The prompt 'λ' is visible at the bottom.

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.

Time: 94 ms, Memory: 4.00 MB
OK (1 test, 1 assertion)

C:\laragon\www\tests\5
λ |
```

PHPUnit  
1 / 1 (100%)

## ExampleTest.php

```
<?php

use PHPUnit\Framework\TestCase;


class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }

    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }
}
```

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..
Time: 99 ms, Memory: 4.00 MB
OK (2 tests, 2 assertions)

C:\laragon\www\tests\5
λ |
```



### ExampleTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }

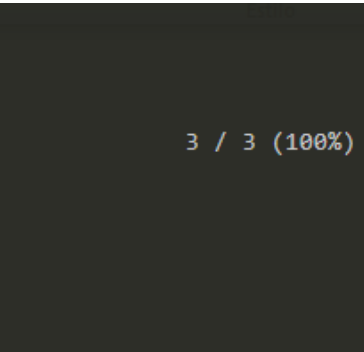
    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }

    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }
}
```

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

...
Time: 101 ms, Memory: 4.00 MB
OK (3 tests, 3 assertions)

C:\laragon\www\tests\5
λ |
```



## ExampleTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }

    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }

    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }

    public function testSame()
    {
        $soma = 5 + 5;
        $this->assertSame($soma, 10);
    }
}
```

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

....                                                                4 / 4 (100%)

Time: 109 ms, Memory: 4.00 MB

OK (4 tests, 4 assertions)
C:\laragon\www\tests\5
λ |
```

## ExampleTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }

    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }

    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }

    public function testSame()
    {
        $soma = 5 + 5;
        $this->assertSame($soma, 10);
    }

    public function testArray()
    {
        $this->assertIsArray(['Roberto', 'Paulo']);
    }
}
```

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.....                                     5 / 5 (100%)
Time: 122 ms, Memory: 4.00 MB
OK (5 tests, 5 assertions)
C:\laragon\www\tests\5
λ |
```

## ExampleTest.php

```
<?php

use PHPUnit\Framework\TestCase;

class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }
    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }
    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }
    public function testSame()
    {
        $soma = 5 + 5;
        $this->assertSame($soma, 10);
    }
    public function testArray()
    {
        $this->assertIsArray(['Roberto', 'Paulo']);
    }
    public function testBool()
    {
        $this->assertIsBool(true);
    }
}
```

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.
```

```
.....
```

```
Time: 107 ms, Memory: 4.00 MB
```

```
OK (6 tests, 6 assertions)
```

```
C:\laragon\www\tests\5
```

```
λ |
```

```
PHPUnit
6 / 6 (100%)
```

```
PHPUnit
4 - 5
PHPUnit
4 - 5
```

## ExampleTest.php

```
<?php
```

```
use PHPUnit\Framework\TestCase;
```

```
class ExampleTest extends TestCase
```

```
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }
    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }
    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }
    public function testSame()
    {
        $soma = 5 + 5;
        $this->assertSame($soma, 10);
    }
    public function testArray()
    {
        $this->assertIsArray(['Roberto', 'Paulo']);
    }
    public function testBool()
    {
        $this->assertIsBool(true);
    }
    public function testInt()
    {
        $this->assertIsInt(7);
    }
}
```

```
C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.....                                     7 / 7 (100%)

Time: 103 ms, Memory: 4.00 MB

OK (7 tests, 7 assertions)

C:\laragon\www\tests\5
λ |
```

## ExampleTest.php

```
<?php
```

```
use PHPUnit\Framework\TestCase;
```

```
class ExampleTest extends TestCase
```

```
{
```

```
    public function testTrue()
```

```
    {
```

```
        $this->assertTrue(true);
```

```
    }
```

```
    public function testFalse()
```

```
    {
```

```
        $var1 = false;
```

```
        $this->assertFalse($var1);
```

```
    }
```

```
    public function testEquals()
```

```
    {
```

```
        $soma = 5 + 5;
```

```
        $this->assertEquals($soma, "10");
```

```
    }
```

```
    public function testSame()
```

```
    {
```

```
        $soma = 5 + 5;
```

```
        $this->assertSame($soma, 10);
```

```
    }
```

```
    public function testArray()
```

```
    {
```

```
        $this->assertIsArray(['Roberto', 'Paulo']);
```

```
    }
```

```
    public function testBool()
```

```
    {
```

```
        $this->assertIsBool(true);
```

```
    }
```

```
    public function testInt()
```

```
    {
```

```
        $this->assertIsInt(7);
```

```
    }
```

```
    public function testString()
```

```
    {
```

```
        $this->assertIsString("Texto");
```

```
    }
```

```
}
```



```

C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..... $this->assertIsInt(7); 8 / 8 (100%)

Time: 106 ms, Memory: 4.00 MB
OK (8 tests, 8 assertions)
C:\laragon\www\tests\5
λ |

```

### ExampleTest.php

```
<?php
```

```
use PHPUnit\Framework\TestCase;
```

```
class ExampleTest extends TestCase
```

```

{
    public function testTrue()
    {
        $this->assertTrue(true);
    }

    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }

    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }

    public function testSame()
    {
        $soma = 5 + 5;
        $this->assertSame($soma, 10);
    }

    public function testArray()
    {
        $this->assertIsArray(['Roberto', 'Paulo']);
    }
}

```

```

public function testBool()
{
    $this->assertIsBool(true);
}

public function testInt()
{
    $this->assertIsInt(7);
}

public function testString()
{
    $this->assertIsString("Texto");
}

public function testEmpty()
{
    $this->assertEmpty([]);
}
}

```

```

C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..... $this->assertIsInt(7); 9 / 9 (100%)

Time: 109 ms, Memory: 4.00 MB
OK (9 tests, 9 assertions)
C:\laragon\www\tests\5
λ

```

### ExampleTest.php

```

<?php

use PHPUnit\Framework\TestCase;

class ExampleTest extends TestCase
{
    public function testTrue()
    {
        $this->assertTrue(true);
    }
    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }
}

```

```

public function testEquals()
{
    $soma = 5 + 5;
    $this->assertEquals($soma, "10");
}
public function testSame()
{
    $soma = 5 + 5;
    $this->assertSame($soma, 10);
}
public function testArray()
{
    $this->assertIsArray(['Roberto', 'Paulo']);
}
public function testBool()
{
    $this->assertIsBool(true);
}
public function testInt()
{
    $this->assertIsInt(7);
}
public function testString()
{
    $this->assertIsString("Texto");
}
public function testEmpty()
{
    $this->assertEmpty([]);
}
public function testCount()
{
    $this->assertCount(2, ['Roberto', 'Paulo']);
}
}

```

```

C:\laragon\www\tests\5
λ vendor\bin\phpunit --action testEmpty()
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.....                                     10 / 10 (100%)

Time: 114 ms, Memory: 4.00 MB

OK (10 tests, 10 assertions)
C:\laragon\www\tests\5
λ

```

## ExampleTest.php

```
<?php
```

```
use PHPUnit\Framework\TestCase;
```

```
class ExampleTest extends TestCase
```

```
{
```

```
    public function testTrue()
    {
        $this->assertTrue(true);
    }

```

```
    public function testFalse()
    {
        $var1 = false;
        $this->assertFalse($var1);
    }

```

```
    public function testEquals()
    {
        $soma = 5 + 5;
        $this->assertEquals($soma, "10");
    }

```

```
    public function testSame()
    {
        $soma = 5 + 5;
        $this->assertSame($soma, 10);
    }

```

```
    public function testArray()
    {
        $this->assertIsArray(['Roberto', 'Paulo']);
    }

```

```
    public function testBool()
    {
        $this->assertIsBool(true);
    }

```

```
    public function testInt()
    {
        $this->assertIsInt(7);
    }

```

```
    public function testString()
    {
        $this->assertIsString("Texto");
    }

```

```

public function testEmpty()
{
    $this->assertEmpty([]);
}

public function testCount()
{
    $this->assertCount(2, ['Roberto', 'Paulo']);
}

public function testHasKey()
{
    $this->assertArrayHasKey('cor', ['cor' => 'azul']);
}
}

```

```

C:\laragon\www\tests\5
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.....                                                    11 / 11 (100%)

Time: 251 ms, Memory: 4.00 MB

OK (11 tests, 11 assertions)
public function testEmpty()
C:\laragon\www\tests\5
λ |
    $this->assertEmpty([]);

```

## Aula 07 - Ciclo do TDD

### Vermelho, Verde, Refactor

1. Primeiro se cria a prova sem o código => vermelho
2. Depois se cria o código => verde
3. Depois se faz as melhorias no código => refactor

Crie a subpasta **6** (exercício 6) e dentro dela instale o **PHPUnit**.

`composer require --dev phpunit/phpunit ^8`

```
C:\laragon\www\tests\6
λ composer require --dev phpunit/phpunit ^8
```

Dentro da subpasta **6** crie as seguintes subpastas:

- **app**
- **tests**

Crie o arquivo **phpunit.xml**

#### **phpunit.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<phpunit bootstrap="vendor/autoload.php"
  colors="true">
  <testsuites>
    <testsuite name="Application Test Suite">
      <directory suffix="Test.php">./tests</directory>
    </testsuite>
  </testsuites>
</phpunit>
```

Altere o arquivo **composer.json** para:

### composer.json

```
{
  "require-dev": {
    "phpunit/phpunit": "8"
  },
  "autoload": {
    "psr-4": {
      "App\\": "app"
    }
  }
}
```

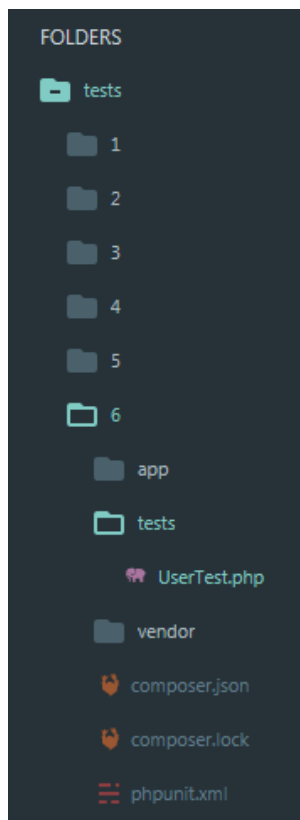
Entre com o comando:

composer dump-autoload

```
C:\laragon\www\tests\6
λ composer dump-autoload
Generated autoload files containing 549 classes

C:\laragon\www\tests\6
λ |
```

Dentro da subpasta **tests**, crie o arquivo (classe) **UserTest.php**



## UserTest.php

```
<?php
```

```
use PHPUnit\Framework\TestCase;
```

```
class UserTest extends TestCase
```

```
{
```

```
}
```



## Aula 08 - TDD - Test de setName e getName

## Etapa Red - Criando a prova sem o código

## UserTest.php

```
<?php

use PHPUnit\Framework\TestCase;
use App\User;

class UserTest extends TestCase
{
    public function test_i_can_register_name()
    {
        $user = new User;
        $user->setName("Italo");

        $this->assertEquals($user->getName(), "Italo");
    }
}
```

```
C:\laragon\www\tests\6
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

E

Time: 1.38 seconds, Memory: 4.00 MB

There was 1 error:

1) UserTest::test_i_can_register_name
Error: Class 'App\User' not found

C:\laragon\www\tests\6\tests\UserTest.php:10

ERRORS!
Tests: 1, Assertions: 0, Errors: 1.

C:\laragon\www\tests\6
λ |
```

## Etapa Verde - Criando o código

Na subpasta **app** crie o arquivo (classe) **User.php**

### User.php

```
<?php

namespace App;

class User
{
    protected $name;

    public function setName($name)
    {
        $this->name = $name;
    }

    public function getName()
    {
        return $this->name;
    }
}
```

```
C:\laragon\www\tests\6
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.                                                                1 / 1 (100%)

Time: 168 ms, Memory: 4.00 MB

OK (1 test, 1 assertion)

C:\laragon\www\tests\6
λ |
```

## Aula 09 - TDD - Test setName e getLastName

### Etapa Red - Criando a prova sem o código

#### UserTest.php

```
<?php

use PHPUnit\Framework\TestCase;
use App\User;

class UserTest extends TestCase
{
    public function test_i_can_register_name()
    {
        $user = new User;
        $user->setName("Italo");

        $this->assertEquals($user->getName(), "Italo");
    }

    public function test_i_can_register_lastname()
    {
        $user = new User;
        $user->setLastName("Morales");

        $this->assertEquals($user->getLastName(), "Morales");
    }
}
```

```
C:\laragon\www\tests\6
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

.E                                                                    2 / 2 (100%)

Time: 122 ms, Memory: 4.00 MB

There was 1 error:

1) UserTest::test_i_can_register_lastname
Error: Call to undefined method App\User::setLastName()

C:\laragon\www\tests\6\tests\UserTest.php:19

ERRORS!
Tests: 2, Assertions: 1, Errors: 1.

C:\laragon\www\tests\6
λ |
```

## Etapa Verde - Criando o código

### User.php

```
<?php

namespace App;

class User
{
    protected $name;
    protected $lastname;

    public function setName($name)
    {
        $this->name = $name;
    }

    public function getName()
    {
        return $this->name;
    }

    public function setLastName($lastname)
    {
        $this->lastname = $lastname;
    }

    public function getLastName()
    {
        return $this->lastname;
    }
}
```

```
C:\laragon\www\tests\6
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.

..                                                                2 / 2 (100%)

Time: 101 ms, Memory: 4.00 MB
OK (2 tests, 2 assertions)
C:\laragon\www\tests\6
λ |
```

## Etapa Refactory

### UserTest.php

```
<?php

use PHPUnit\Framework\TestCase;
use App\User;

class UserTest extends TestCase
{
    protected $user;

    public function setUp () : void
    {
        $this->user = new User;
    }

    public function test_i_can_register_name()
    {
        $this->user->setName("Italo");
        $this->assertEquals($this->user->getName(), "Italo");
    }

    public function test_i_can_register_lastname()
    {
        $this->user->setLastName("Morales");
        $this->assertEquals($this->user->getLastName(), "Morales");
    }
}
```

```
C:\laragon\www\tests\6
λ vendor\bin\phpunit
PHPUnit 8.0.0 by Sebastian Bergmann and contributors.
```

```
..
```

```
Time: 101 ms, Memory: 4.00 MB
```

```
OK (2 tests, 2 assertions)
```

```
C:\laragon\www\tests\6
```

```
λ |
```

```
PHPUnit
2 / 2 (100%)
```

