



O presente documento visa instruir em forma de passo a passo um usuário na utilização da ferramenta de automação de testes conhecida por **PHPUnit**, focada em testes unitários na linguagem de programação e desenvolvimento web PHP. Ao finalizar o tutorial, o usuário deve ser capaz de instalar o Framework, habilitar extensões e criar suítes de testes com a ferramenta.

1 Instalação

Para poder rodar o PHPUnit, é necessário primeiro instalar o PHP em sua máquina, você pode realizar esse procedimento acessando o site: <https://www.php.net/downloads.php>. Neste tutorial estaremos utilizando a versão 8.2.10 do PHP versão Non Thread Safe. Caso esteja utilizando o sistema operacional Windows, você pode acessar o link de Download clicando na opção Windows downloads e baixar o arquivo zip.

A screenshot of the PHP 8.2.10 download page. The page has a dark blue header with the PHP logo and navigation links: Downloads, Documentation, Get Involved, Help, and php 8.2. The main content area is light gray and titled "Current Stable PHP 8.2.10 (Changelog)". It lists three download options: php-8.2.10.tar.gz (18,657Kb), php-8.2.10.tar.bz2 (14,949Kb), and php-8.2.10.tar.xz (11,759Kb). Each option includes a SHA256 hash and a "31 A" label. A red arrow points to the "Windows downloads" link at the bottom of the list. Below the list is a link for "GPG Keys for PHP 8.2".

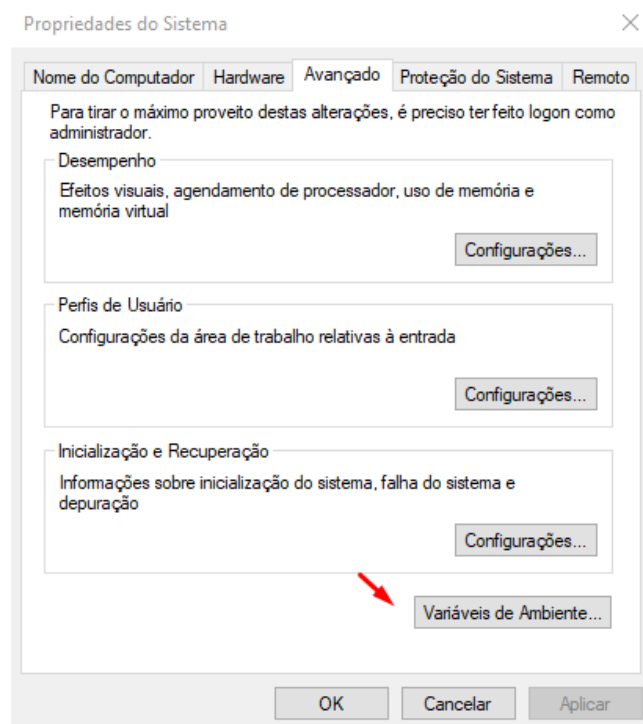
Download Link	Size	SHA256 Hash	Label
php-8.2.10.tar.gz (sig)	[18,657Kb]	sha256: 7e3e277d6eab652616f90bc7c75991179c0512953933ceba27496fb5514f7e78	31 A
php-8.2.10.tar.bz2 (sig)	[14,949Kb]	sha256: cc9834e8f1b613d7677af8843c3651e9829abca8ebfe9079251d0d85d9a0aa3e	31 A
php-8.2.10.tar.xz (sig)	[11,759Kb]	sha256: 561dc4acd5386e47f25be76f2c8df6ae854756469159248313bcf276e282fbb3	31 A
Windows downloads			

[GPG Keys for PHP 8.2](#)

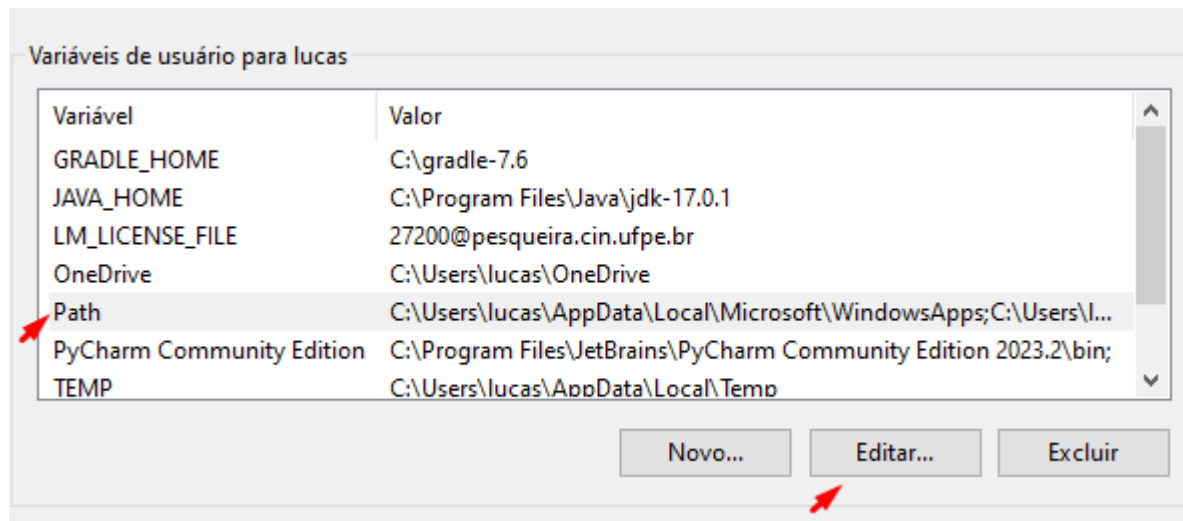
VS16 x64 Non Thread Safe (2023-Aug-30 10:16:12)

- [Zip](#) [30.24MB]
sha256: 2b6006ca0af796aca31565130d592ede74abf9d8d1b1da36bac6f748a20f9eb2
- [Debug Pack](#) [24.52MB]
sha256: 6f1477cb72915462ee49dfc1d6be09f021714b783786b457416eede90ce5ba16
- [Development package \(SDK to develop PHP extensions\)](#) [1.24MB]
sha256: a397114171adeb3cad1bb99c33e41a1f958d1feac19fc5b544fb6373a3e53947

Com o arquivo baixado você pode extrair o zip em uma pasta de preferência, recomenda-se realizar essa extração diretamente no diretório C: do Windows. Seguindo essa recomendação, o caminho de instalação do seu PHP deve ser algo semelhante a: **C:\php**. Agora você deve inserir esse caminho na variável de ambiente PATH do seu sistema operacional para que o terminal possa reconhecer os comandos. Você deve fazer isso indo no menu do seu Windows e digitar variáveis de ambiente, deve aparecer uma opção para editá-las como no print abaixo:



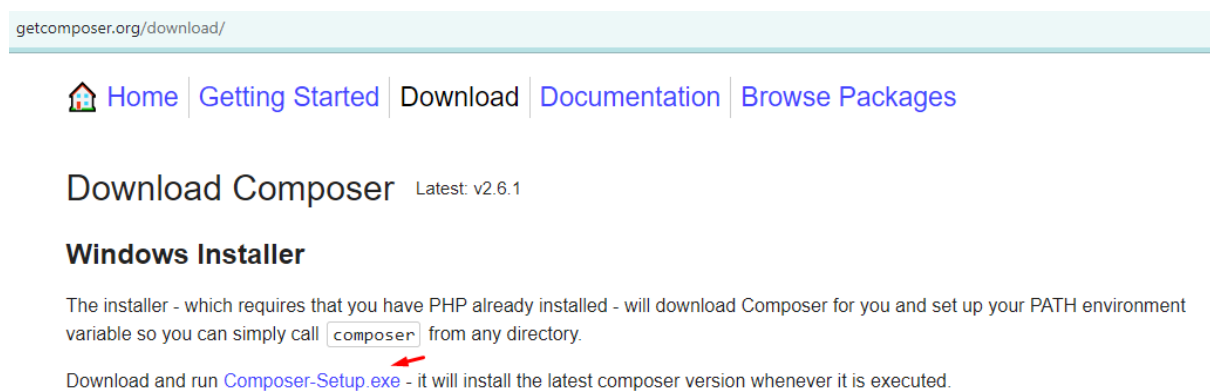
Você seleciona a variável PATH e clique em Editar para adicionar o novo caminho:



Depois de adicionar o C:\php você pode ir clicando em OK e basta reiniciar o seu prompt de comando para que os comandos sejam reconhecidos.

Após instalar o PHP, é também necessário baixar o Composer que é uma ferramenta para gerenciamento de pacotes em PHP com o objetivo de facilitar a administração das bibliotecas de um projeto, instalando-as e atualizando-as automaticamente.

Para utilizar o composer, basta acessar o link <https://getcomposer.org/download/> e baixar o arquivo executável Composer-Setup.exe.



Basta executar o arquivo baixado que ele instalará a versão mais recente do composer e já adicionará automaticamente na variável PATH o caminho de instalação da ferramenta.

2 Começando com o PHPUnit

Com o composer instalado você pode criar uma nova pasta para utilizar um projeto PHP com PHPUnit. Dentro da nova pasta criada, rode o comando:

```
composer require --dev --prefer-source phpunit/phpunit
```

Após instalado todas as dependências do PHPUnit, você pode rodar o seguinte comando:

```
vendor/bin/phpunit
```

Ele irá mostrar um menu inicial com diversas opções para utilização da ferramenta como parâmetros de configuração, seleção de testes, execução de testes, relatório de testes, log de testes e até code coverage. Esse menu deve aparecer em seu terminal de forma semelhante ao print abaixo:

```
PS C:\Users\lucas\Desktop\PHPUnit> vendor/bin/phpunit
PHPUnit 10.3.2 by Sebastian Bergmann and contributors.

Usage:
  phpunit [options] UnitTest.php
  phpunit [options] <directory>

Configuration:

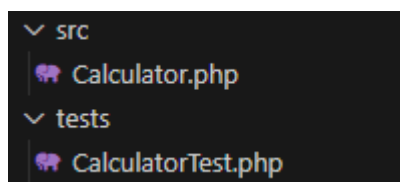
  --bootstrap <file>          A PHP script that is included before the tests run
  -c|--configuration <file>  Read configuration from XML file
  --no-configuration          Ignore default configuration file (phpunit.xml)
  --no-extensions              Do not load PHPUnit extensions
  --include-path <path(s)>    Prepend PHP's include_path with given path(s)
  -d <key[=value]>            Sets a php.ini value
  --cache-directory <dir>    Specify cache directory
  --generate-configuration    Generate configuration file with suggested settings
  --migrate-configuration     Migrate configuration file to current format

Selection:

  --list-suites               List available test suites
  --testsuite <name>         Only run tests from the specified test suite(s)
  --exclude-testsuite <name> Exclude tests from the specified test suite(s)
  --list-groups               List available test groups
  --group <name>              Only run tests from the specified group(s)
  --exclude-group <name>     Exclude tests from the specified group(s)
  --covers <name>             Only run tests that intend to cover <name>
```

3 Codando com PHPUnit

Na estrutura de diretórios de seu projeto, é recomendado criar uma pasta src para deixar o código que será testado e uma pasta somente para os testes, como no print abaixo:



Estaremos utilizando a classe Calculator para estar criando testes unitários com o PHPUnit. O código completo você poderá ter acesso através do seguinte diretório público no Github: https://github.com/lucascin/PHPUnit_tutorial

Mas apenas para ilustrar no presente documento, aqui temos uma função da classe Calculator para a realização da soma da calculadora na linguagem PHP.

```
<?php

class Calculator {

    protected $result;

    public function add($a, $b) {
        return $a + $b;
    }
}

?>
```

Na pasta tests, criamos a suíte de testes CalculatorTest.php.

```
<?php

use PHPUnit\Framework\TestCase;
require_once 'src/Calculator.php';
```

Essas são as linhas iniciais do código CalculatorTest que importa a classe TestCase do PHPUnit para a criação dos testes e faz a requisição do arquivo com a classe que será testada na suíte.

```
class CalculatorTest extends TestCase {

    private $calculator;

    protected function setUp(): void {
        $this->calculator = new Calculator();
    }

    public function testAddition() {
        $result = $this->calculator->add(2, 3);
        $this->assertEquals(5, $result);
    }

}

?>
```

Essa é a sintaxe para a criação de um teste unitário utilizando o framework PHPUnit. Com a variável \$result declarada você pode realizar chamadas para o objeto calculator declarado no setUp e utilizar as funções da classe Calculator.

Para rodar a suíte de testes, basta ir no terminal na pasta raiz do seu projeto e rodar o seguinte comando vendor/bin/phpunit + o nome da sua suíte de teste.

No caso do exemplo do tutorial, o comando seria algo como:

```
vendor/bin/phpunit tests/CalculatorTest.php
```

Com isso, você poderá verificar se o teste passou ou falhou de acordo com os parâmetros definidos no caso de teste.

```
PS C:\Users\lucas\Desktop\PHPUnit> vendor/bin/phpunit tests/CalculatorTest.php
PHPUnit 10.3.2 by Sebastian Bergmann and contributors.

.                                                                    1 / 1 (100%)

Time: 00:00.071, Memory: 6.00 MB
```

4 Code Coverage

O PHPUnit possui uma extensão com suporte a coverage chamada Xdebug, nessa seção estaremos aprendendo como habilitar a extensão e utilizar na suíte.

Para baixar o Xdebug, você pode ir ao site: <https://xdebug.org/download>

Selecione o arquivo de acordo com a versão de PHP que você instalou.

De acordo com o passo a passo do tutorial, o PHP instalado em sua máquina foi a versão 8.2 sem o Thread Safe. A versão correta a se baixar do Xdebug deve ser a indicada no print:

Latest Release

Xdebug 3.2.2

Release date: 2023-07-14

- Linux, macOS:

[source](#)

- Windows binaries:

[PHP 8.0 VS16 \(64 bit\)](#)

[PHP 8.0 VS16 TS \(64 bit\)](#)

[PHP 8.1 VS16 \(64 bit\)](#)

[PHP 8.1 VS16 TS \(64 bit\)](#)

[PHP 8.2 VS16 \(64 bit\)](#)

[PHP 8.2 VS16 TS \(64 bit\)](#)

Copie esse arquivo baixado e cole no diretório ext na pasta de instalação do seu php, o caminho, seguindo o tutorial deve ser equivalente a: C:\php\ext

Recomenda-se que você renomeie o arquivo somente para o nome da extensão: php_xdebug.dll

php > ext		Pesquisar em ext	
Nome	Data de modificação	Tipo	
php_pdo_firebird.dll	30/08/2023 07:15	Extensão de aplica	
php_pdo_mysql.dll	30/08/2023 07:15	Extensão de aplica	
php_pdo_oci.dll	30/08/2023 07:15	Extensão de aplica	
php_pdo_odbc.dll	30/08/2023 07:15	Extensão de aplica	
php_pdo_pgsql.dll	30/08/2023 07:15	Extensão de aplica	
php_pdo_sqlite.dll	30/08/2023 07:15	Extensão de aplica	
php_pgsql.dll	30/08/2023 07:15	Extensão de aplica	
php_shmop.dll	30/08/2023 07:15	Extensão de aplica	
php_snmp.dll	30/08/2023 07:15	Extensão de aplica	
php_soap.dll	30/08/2023 07:15	Extensão de aplica	
php_sockets.dll	30/08/2023 07:15	Extensão de aplica	
php_sodium.dll	30/08/2023 07:15	Extensão de aplica	
php_sqlite3.dll	30/08/2023 07:15	Extensão de aplica	
php_sysvshm.dll	30/08/2023 07:15	Extensão de aplica	
php_tidy.dll	30/08/2023 07:15	Extensão de aplica	
php_xdebug.dll	01/09/2023 01:14	Extensão de aplica	
php_xsl.dll	30/08/2023 07:15	Extensão de aplica	
php zend_test.dll	30/08/2023 07:15	Extensão de aplica	
php_zip.dll	30/08/2023 07:15	Extensão de aplica	

Agora com o Xdebug presente na sua pasta, você pode gerar um arquivo padrão de configuração com o comando: `phpunit --generate-configuration`

Basta apertar enter nas perguntas defaults que serão acionadas no terminal durante a geração do arquivo padrão de configuração.

Após esse comando, você deve ir no arquivo `php.ini` localizado na pasta de instalação do seu PHP para habilitar o Xdebug.

Localize essa sessão no arquivo `php.ini`:

```
;extension=soap
;extension=sockets
;extension=sodium
;extension=sqlite3
;extension=tidy
;extension=xsl
;extension=zip
```

Insira agora as seguintes linhas abaixo:

```
zend_extension=php_xdebug.dll
xdebug.mode=develop,debug,coverage
```

Salve o arquivo e vamos para o implementação no código.

Para obter os dados de coverage em um relatório, é necessário definir no código qual função o teste irá cobrir. Abaixo você pode visualizar a sintaxe padrão para definir @covers no PHPUnit.

```
/**
 * @covers Calculator::add
 */

public function testAddition() {
    $result = $this->calculator->add(2, 3);
    $this->assertEquals(5, $result);
}
```

@covers Indica que o teste abaixo irá cobrir a função add da classe Calculator. Definindo os covers das funções agora podemos ir para o terminal testar a funcionalidade.

No terminal na pasta raiz, você pode rodar agora o comando:

```
vendor/bin/phpunit --coverage-html PHPUnit_coverage_report tests/CalculatorTest.php
```













Onde você irá rodar a suíte de testes e determina a geração de um arquivo de relatório da cobertura do código com o parâmetro --coverage-html e determina que esse relatório deve ser armazenado na pasta PHPUnit_coverage_report.

PHPUnit > PHPUnit_coverage_report

Nome	Data de modificação	Tipo	Tamanho
_css	01/09/2023 01:55	Pasta de arquivos	
_icons	01/09/2023 01:55	Pasta de arquivos	
_js	01/09/2023 01:55	Pasta de arquivos	
Calculator.php	01/09/2023 01:55	Chrome HTML Do...	22 KB
dashboard	01/09/2023 01:55	Chrome HTML Do...	8 KB
index	01/09/2023 01:55	Chrome HTML Do...	6 KB

Basta acessar a classe Calculator.php e verificar o relatório de cobertura no navegador.

C:\Users\lucas\Desktop\PHPUnit\src / Calculator.php

	Code Coverage								
	Lines			Functions and Methods				Classes and Traits	
Total		100.00%	8 / 8		100.00%	4 / 4	CRAP		100.00% 1 / 1
Calculator		100.00%	8 / 8		100.00%	4 / 4	5		100.00% 1 / 1
add		100.00%	1 / 1		100.00%	1 / 1	1		
subtract		100.00%	1 / 1		100.00%	1 / 1	1		
multiply		100.00%	1 / 1		100.00%	1 / 1	1		
divide		100.00%	5 / 5		100.00%	1 / 1	2		

```
1 <?php
2
3 class Calculator {
4
5     protected $result;
6     public function add($a, $b) {
7         return $a + $b;
8     }
9
10    public function subtract($a, $b) {
11        return $a - $b;
12    }
13 }
```

Como pode ver, existem mais funções implementadas na classe Calculator e mais testes criados na suíte TestCalculator do que os descritos na seção 3 do tutorial, para ter acesso a toda a suíte de testes e a classe Calculator completa, basta acessar o repositório do Github: https://github.com/lucascin/PHPUnit_tutorial

Link para vídeo utilizando a ferramenta:

https://drive.google.com/file/d/1KA_3TPnMHAyn610HaSe-Y2EF8qEGgYmy/view?usp=sharing