Getting Started

Architecture Concepts

The Basics

Frontend

Security

Digging Deeper

Database

Eloquent ORM

Testing

Getting Started

HTTP Tests

Console Tests

Browser Tests

Database

Mocking

Official Packages



Deploy Laravel and PHP applications on DigitalOcean, Linode, &

ADS VIA CARBON

Q Search Docs

6.x

Testing: Getting Started

- # Introduction
- # Environment
- # Creating & Running Tests

Introduction

Laravel is built with testing in mind. In fact, support for testing with PHPUnit is included out of the box and a phpunit.xml file is already set up for your application. The framework also ships with convenient helper methods that allow you to expressively test your applications.

By default, your application's tests directory contains two directories: Feature and Unit. Unit tests are tests that focus on a very small, isolated portion of your code. In fact, most unit tests probably focus on a single method. Feature tests may test a larger portion of your code, including how several objects interact with each other or even a full HTTP request to a JSON endpoint.

An ExampleTest.php file is provided in both the Feature and Unit test directories. After installing a new Laravel application, run phpunit on the command line to run your tests.

Environment

When running tests via <a href="https://pubmer.com/physics.com/phy

You are free to define other testing environment configuration values as necessary. The testing environment variables may be configured in the phpunit.xml file, but make sure to clear your configuration cache using the config:clear Artisan command before running your tests!

In addition, you may create a .env.testing file in the root of your project. This file will override the .env file when running PHPUnit tests or executing Artisan commands with the --env=testing option.

Creating & Running Tests

To create a new test case, use the make:test Artisan command:

```
// Create a test in the Feature directory...
php artisan make:test UserTest

// Create a test in the Unit directory...
php artisan make:test UserTest --unit
```

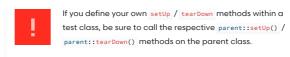
Once the test has been generated, you may define test methods as you normally would using PHPUnit. To run your tests, execute the phpunit command from your terminal:

```
c?php

namespace Tests\Unit;

use Illuminate\Foundation\Testing\RefreshDatabase;
use Tests\TestCase;

class ExampleTest extends TestCase
{
    /**
    * A basic test example.
    *
    * @return void
    */
    public function testBasicTest()
    {
        $this->assertTrue(true);
    }
}
```



Become a Laravel Partner

Laravel Partners are elite shops providing top-notch Laravel development and consulting. Each of our partners can help you craft a beautiful, well-architected project.

Our Partners

Larave

Highlights

esources

Partners

Ecosysten

arayel is a web application framework with expressive

Release Notes Vehikl Laracasts Vapor Getting Started Laravel News Tighten Co. Forge Routing Kirschbaum Laracon Envoyer Blade Templates Laracon EU Byte 5 Horizon Authentication Laracon AU 64Robots Lumen Cubet Authorization Jobs Nova Artisan Console Certification DevSquad Echo Database Forums Valet Eloquent ORM Cyber-Duck Mix Testing ABOUT YOU Spark Become A Partner

Cashier Homestead Dusk Passport Scout

Socialite

elegant syntax. We believe development must be an enjoyable and creative experience to be truly fulfilling. Laravel attempts to take the pain out of development by easing common tasks used in most web projects.







