

PHPUnit

Pruebas unitarias para PHP



Gestor de dependencias para PHP

```
1 {  
2   "require": {  
3     "red5g/auth": "2.0"  
4   }  
5 }
```

INDICANDO DEPENDENCIAS DEL PROYECTO

composer.json


```
1 {  
2   "require": {  
3     "illuminate/database": "v4.2.9"  
4   },  
5   "require-dev": {  
6     "phpunit/phpunit": "4.6.0"  
7   }  
8 }
```

REQUIRIENDO ELOQUENT Y PHPUNIT

require-dev nos permite separar los entornos

```
> composer.phar install  
> composer.phar install --prefer-dist  
> composer.phar install --prefer-source  
> composer.phar update
```

COMANDOS DE INSTALACIÓN

Dos formas o fuentes de instalación

 vendor

 autoload.php

 composer.lock

ARCHIVOS GENERADOS POR COMPOSER

Recomendable no excluir composer.lock de git

```
1 <?php
2
3 function __autoload($nombre_clase)
4 {
5     include $nombre_clase . '.php';
6 }
7
8 $obj = new MiClase1();
9 $obj2 = new MiClase2();
```

FUNCIÓN AUTOLOAD

__autoload() podría quedar obsoleta.

Este ejemplo ilustra la funcionalidad de la autocarga de clases.


```
1 {
2   "require": {
3     "illuminate/database": "v4.2.9"
4   },
5   "require-dev": {
6     "phpunit/phpunit": "4.6.0"
7   },
8   "autoload": {
9     "psr-4": {
10       "Red5g\\": "app",
11       "Red5g\\Models\\": "app/models"
12     }
13   }
14 }
```

AUTOLOAD USANDO COMPOSER Y PSR-4

Las rutas son relativas a composer.json


```
1 <?php
2
3 require_once 'vendor/autoload.php';
4
5 use Illuminate\Database\Eloquent\Model;
6 use Red5g\User;
7
```

INCLUIR EL ARCHIVO AUTOLOAD

Ejemplo de uso

HABLEMOS DE PRUEBAS



PHPUnit

- Utiliza **aserciones** para verificar que el comportamiento de una *unidad* de código es el esperado.
- **Aserción** (RAE): Proposición en que se afirma o da por cierto algo.

CONVENCIONES

- Estructura y nombre de archivos: Debe imitar la estructura y nombre de los archivos del proyecto.
- Las clases deben extender a *PHPUnit_Framework_TestCase* y el nombre debe terminar con la palabra *Test*. e.g: *FormatTest*.
- Nombres de tests: deben empezar con la palabra *test* (en minúscula) y deben ser tan descriptivos como sea posible... *testDebitoIgualCreditoComprobanteEgreso()*.
- Los tests deben ser métodos públicos.

```
6
7 class TrueTest extends \PHPUnit_Framework_TestCase
8 {
9     public function testTrueIsTrue()
10    {
11        $foo = true;
12        $this->assertTrue($foo);
13    }
14
```

ASSERT

Ejemplo básico con PHPUnit

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <phpunit colors="true">
3     <testsuites>
4         <testsuite name="Application Test Suite">
5             <directory>./app/Test/</directory>
6         </testsuite>
7     </testsuites>
8 </phpunit>
```

phpunit.xml

Archivo de configuración


```
> vendor/bin/phpunit
```

CORRER TESTS

Ubicarse en directorio raíz del proyecto

HAGAMOS ALGUNOS TESTS



“Whenever you are tempted to type something into a print statement or a debugger expression, write it as a test instead”

– Martin Fowler



<https://github.com/jtrezza/PHPUnitRed5G>

Repositorio de GitHub

GRACIAS