

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

Control de autenticación y
 acceso (Spring Security)
 UDDI
 Web Services
 Rest Services
 Social SSO
 SSO (Cas)

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)


[Entra er](#)

[Inicio](#)
[Quiénes somos](#)
[Tutoriales](#)
[Formación](#)
[Comparador de salarios](#)
[Nuestro libro](#)
[Charlas](#)

» Estás en: [Inicio](#) [Tutoriales](#) [TDD con PHP gracias a PHPUnit](#)



[Alejandro Pérez García](#)

Alejandro es socio fundador de Autentia y nuestro experto en J2EE, Linux y optimización de aplicaciones empresariales.

Ingeniero en Informática y Certified ScrumMaster

Si te gusta lo que ves, puedes contratarle para darte ayuda con soporte experto, impartir **cursos presenciales** en tu empresa o para que **realicemos tus proyectos como factoría** (Madrid). Puedes encontrarme en [Autentia](#): Ofrecemos servicios de soporte a desarrollo, factoría y formación.

[Ver todos los tutoriales del autor](#)

TE ASESORAMOS GRATIS
900 10 27 29

O busca en: **TopFormacion.com**
Cursos, masters, oposiciones...

Fecha de publicación del tutorial: 2012-03-05

Tutorial visitado 189 veces [Descargar en PDF](#)

Como hacer TDD con PHP gracias a PHPUnit

Creación: 04-03-2012

Índice de contenidos

- [1. Introducción](#)
- [2. Entorno](#)
- [3. Como instalar PHP](#)
- [4. Como instalar PEAR](#)
- [5. Como instalar PHPUnit](#)
- [6. Un ejemplo de uso de PHPUnit](#)
- [7. Como instalar XDebug](#)
- [4. Conclusiones](#)
- [5. Sobre el autor](#)

1. Introducción

La vida da muchas vueltas y en estos días me he tenido que poner a hacer algunas cosillas en PHP. Como no tenía ni idea lo primero que hice fue leerme el manual, y ya puedo afirmar "Ya sé PHP!". Bueno, en realidad sigo sin tener ni idea, pero por lo menos conozco los principios del lenguaje ;)

Ante este panorama lo segundo que hice fue investigar como se puede hacer TDD en PHP. Para mi esto es fundamental, no sólo porque considere que el TDD es imprescindible en si mismo, sino porque teniendo en cuenta mi inexperiencia con PHP, no me atrevo a tocar ni una sola línea de código si no es con un buen test que me proteja de la caída.

Así, en este tutorial, voy a contaros como he configurado mi entorno (Mac OS X) para poder desarrollar en PHP con TDD y pruebas automáticas con [PHPUnit](#); un framework de pruebas unitarias del estilo xUnit, donde incluso tenemos soporte para Stubs, Mocks, análisis de cobertura, y muchas más cosas (de verdad muy recomendable si trabajamos con PHP).

Cat
Aut

Síg



Últi

» Al
las r
Adrr

» X
Myb
Hibe

» Pr
Lag:

» Cu
prep
apai

» jji
trafc
Hist

Últi

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 15' (2.5 GHz Intel i7, 8GB 1333 Mhz DDR3, 256GB Solid State Drive).
- AMD Radeon HD 6770M 1024 MB
- Sistema Operativo: Mac OS X Lion 10.7.3
- PHP 5.3.8
- PEAR 1.9.4
- PHPUnit 3.6.10

3. Como instalar PHP

Con Mac OS X ya tenemos instalado Apache (servidor Web) y PHP, sólo tenemos que activarlo. Para ello editamos `/etc/apache2/httpd.conf`, buscamos la línea:

```
1 | #LoadModule php5 module libexec/apache2/libphp5.so
```

Y le quitamos la #, es decir activamos el módulo de PHP del Apache.

Por defecto el directorio donde el Apache espera encontrar nuestras páginas es `/Library/WebServer/Documents`. Pero esta ruta puede ser poco conveniente para albergar nuestros proyectos, así que en el mismo `httpd.conf` podemos crearnos un *Virtual Host*, para ello añadimos algo como:

```
1 | <Directory "/mis-proyectos/autentia/php/tutorial/php-unit">
2 |     Options FollowSymLinks MultiViews
3 |     AllowOverride All
4 |     Order allow,deny
5 |     Allow from all
6 | </Directory>
7 |
8 | <VirtualHost www.tutorial-phpunit.com>
9 |     ServerName tutorial-phpunit.com
10 |     DocumentRoot "/mis-proyectos/autentia/php/tutorial/php-unit"
11 | </VirtualHost>
```

Donde `/mis-proyectos/autentia/php/tutorial/php-unit` es la ruta donde queremos tener los fuentes de nuestro proyecto PHP.

Nótese también que estamos usando el dominio `www.tutorial-phpunit.com`, este lo tendremos que haber definido en el fichero `/etc/hosts` para que apunte a nuestra máquina:

```
1 | 127.0.0.1 www.tutorial-phpunit.com
```

También vamos a poner el fichero de configuración de PHP, el `php.ini`. Crear este fichero de configuración va a ser muy sencillo porque nuevamente Mac OS X nos lo facilita dándonos uno por defecto, así sólo tendremos que hacer:

```
$ sudo cp /etc/php.ini.default /etc/php.ini
```

Ahora reiniciamos el Apache con el comando:

```
$ sudo /usr/sbin/apachectl restart
```

(también podemos hacerlo desde *System Preferences... -> Sharing -> Web Sharing*)

4. Como instalar PEAR

PEAR, tal como lo definen en su propia página web, es un framework y un sistema de distribución para componentes PHP reutilizables. Básicamente podríamos decir que es un sistema de gestión de paquetes, al estilo de *apt-get* para los paquetes de Debian, o *gem* para las gemas de Ruby, o *Maven* para las librerías de Java.

En esta ocasión lo necesitamos para poder instalar luego PHPUnit.

De nuevo en Mac OS X su instalación es muy sencilla, basta con ejecutar:

```
$ cd /usr/lib/php ; sudo php install-pear-nozlib.phar
```

La salida del comando debe ser algo como:

```
1 | [PEAR] Archive_Tar      - installed: 1.3.7
2 | [PEAR] Console_Getopt  - installed: 1.3.0
3 | [PEAR] Structures_Graph - installed: 1.0.4
```

```

4 [PEAR] XML Util      - installed: 1.2.1
5 [PEAR] PEAR         - installed: 1.9.4
6 Wrote PEAR system config file at: /private/etc/pear.conf
7 You may want to add: /usr/lib/php/pear to your php.ini include path

```

Ahora tal como nos dice la última línea, vamos a modificar el fichero `/etc/php.ini`, para tener en cuenta el directorio donde se ha instalado PEAR. Para ello en dicho fichero buscamos la línea:

```
1 ;include_path = "./:/php/includes"
```

Y la cambiamos por:

```
1 include_path = "./:/usr/lib/php/pear"
```

5. Como instalar PHPUnit

PHPUnit es el estándar de facto para hacer test unitarios en proyectos PHP. Para instalar PHPUnit basta con hacer:

```

$ sudo pear config-set auto_discover 1
$ sudo pear install pear.phpunit.de/PHPUnit

```

La salida de este último comando será algo similar a:

```

1 Attempting to discover channel "pear.phpunit.de"...
2 downloading channel.xml ...
3 Starting to download channel.xml (804 bytes)
4 ...done: 804 bytes
5 Auto-discovered channel "pear.phpunit.de", alias "phpunit", adding to registry
6 Attempting to discover channel "pear.symfony-project.com"...
7 downloading channel.xml ...
8 Starting to download channel.xml (865 bytes)
9 ...done: 865 bytes
10 Auto-discovered channel "pear.symfony-project.com", alias "symfony", adding to registry
11 Did not download optional dependencies: phpunit/PHP_Invoker, use --allddeps to download aut
12 phpunit/PHPUnit can optionally use package "phpunit/PHP_Invoker" (version >= 1.1.0)
13 phpunit/PHP_CodeCoverage can optionally use PHP extension "xdebug" (version >= 2.0.5)
14 downloading PHPUnit-3.6.10.tgz ...
15 Starting to download PHPUnit-3.6.10.tgz (118,595 bytes)
16 ...done: 118,595 bytes
17 downloading File_Iterator-1.3.1.tgz ...
18 Starting to download File_Iterator-1.3.1.tgz (5,157 bytes)
19 ...done: 5,157 bytes
20 downloading Text_Template-1.1.1.tgz ...
21 Starting to download Text_Template-1.1.1.tgz (3,622 bytes)
22 ...done: 3,622 bytes
23 downloading PHP_CodeCoverage-1.1.2.tgz ...
24 Starting to download PHP_CodeCoverage-1.1.2.tgz (132,552 bytes)
25 ...done: 132,552 bytes
26 downloading PHP_Timer-1.0.2.tgz ...
27 Starting to download PHP_Timer-1.0.2.tgz (3,686 bytes)
28 ...done: 3,686 bytes
29 downloading PHPUnit_MockObject-1.1.1.tgz ...
30 Starting to download PHPUnit_MockObject-1.1.1.tgz (19,897 bytes)
31 ...done: 19,897 bytes
32 downloading YAML-1.0.6.tgz ...
33 Starting to download YAML-1.0.6.tgz (10,010 bytes)
34 ...done: 10,010 bytes
35 downloading PHP_TokenStream-1.1.3.tgz ...
36 Starting to download PHP_TokenStream-1.1.3.tgz (9,860 bytes)
37 ...done: 9,860 bytes
38 install ok: channel://pear.phpunit.de/File_Iterator-1.3.1
39 install ok: channel://pear.phpunit.de/Text_Template-1.1.1
40 install ok: channel://pear.phpunit.de/PHP_Timer-1.0.2
41 install ok: channel://pear.symfony-project.com/YAML-1.0.6
42 install ok: channel://pear.phpunit.de/PHP_TokenStream-1.1.3
43 install ok: channel://pear.phpunit.de/PHP_CodeCoverage-1.1.2
44 install ok: channel://pear.phpunit.de/PHPUnit_MockObject-1.1.1
45 install ok: channel://pear.phpunit.de/PHPUnit-3.6.10

```

Podemos instalar más módulos de PHPUnit en función de nuestras necesidades (integración con base de datos, test de Selenium, ...), para ello podemos encontrar más información en [la página de instalación de PHPUnit](#).

6. Un ejemplo de uso de PHPUnit

Una buena forma de organizar nuestros test es separarlos del código de producción y tener un fichero de test por cada clase o fichero php, duplicando la estructura de directorios de producción en el directorio de test, como si se tratara de un espejo. Por ejemplo, la librería [Object Freezer](#) sigue al siguiente estructura:

```

Object                                     Tests
|-- Freezer                               |-- Freezer
|   |-- HashGenerator                     |   |-- HashGenerator

```

```

|-- NonRecursiveSHA1.php
|-- HashGenerator.php
|-- IdGenerator
   |-- UUID.php
   |-- IdGenerator.php
   |-- LazyProxy.php
   |-- Storage
       |-- CouchDB.php
|-- Storage.php
|-- Util.php
|-- Freezer.php

|-- NonRecursiveSHA1Test.php
|-- IdGenerator
   |-- UUIDTest.php
|-- Storage
   |-- WithLazyLoadTest.php
   |-- WithoutLazyLoadTest.php
|-- StorageTest.php
|-- UtilTest.php
|-- FreezerTest.php

```

Nosotros vamos a hacer un ejemplo más sencillo donde vamos a tener un directorio *main* con el código de producción, y un directorio *test* con el código de los test. De esta forma tendremos los ficheros:

```

1  <?php
2  /** /mis-proyectos/autentia/php/tutorial/php-unit/AdictosTutorial.php */
3
4  class AdictosTutorial {
5
6      public function greet() {
7          return 'Hola Adictos Al Trabajo !!!';
8      }
9  }
10
11  ?>

```

```

1  <?php
2  /** /mis-proyectos/autentia/php/tutorial/php-unit/test/AdictosTutorialTest.php */
3
4  require_once 'main/AdictosTutorial.php';
5
6  class AdictosTutorialTest extends PHPUnit_Framework_TestCase {
7
8      public function testReturnGreeting() {
9          $adictos = new AdictosTutorial();
10         $this->assertEquals('Hola Adictos Al Trabajo !!!', $adictos->greet());
11     }
12 }
13
14  ?>

```

Desde el directorio `/mis-proyectos/autentia/php/tutorial/php-unit` ejecutaremos el siguiente comando para lanzar los test:

```
$ phpunit test
```

Y obtendremos la siguiente salida con los resultados de la ejecución de los test (cada `.` representa un test):

```

1  PHPUnit 3.6.10 by Sebastian Bergmann.
2
3  .
4
5  Time: 0 seconds, Memory: 5.50Mb
6
7  OK (1 test, 1 assertion)

```

Esto es sólo la punta del iceberg, así que os recomiendo que leáis la [completísima documentación de PHPUnit](#).

7. Como instalar XDebug

XDebug es una herramienta de depuración y análisis para PHP. En principio la parte de depuración no nos interesa porque nosotros preferimos el camino del TDD ;) pero si nos puede venir muy bien para hacer análisis de rendimiento o para ver la cobertura de nuestros test de PHPUnit.

Para activarlo editamos el fichero `/etc/php.ini` y buscamos la línea:

```
1 ;zend_extension="/usr/lib/php/extensions/no-debug-non-ztsV-20090626/xdebug.so"
```

Le quitamos el `;` para descomentarla y ya estamos listos!

Si vamos a querer hacer análisis de cobertura necesitamos definir nuestra zona horaria, para ello buscamos la línea:

```
1 ;date.timezone =
```

Y pondremos los datos de nuestra zona horaria, por ejemplo para Madrid:

```

1 ; Madrid, Spain, Europe
2 date.timezone = "Europe/Madrid"
3 date.default_latitude = 40.416126
4 date.default_longitude = -3.696706

```

Ahora para lanzar los test y medir la cobertura podemos ejecutar:

```
$ phpunit --coverage-html ./report test
```

Donde `./report` es el directorio donde dejaremos los resultados del análisis de cobertura en formato HTML, y `test` es el directorio donde tenemos los tests.

4. Conclusiones

PHP! Quién dijo miedo?!?!?!! Haciendo TDD y teniendo una buena batería de test, no hay lenguaje que se nos resista!

5. Sobre el autor

Alejandro Pérez García, Ingeniero en Informática (especialidad de Ingeniería del Software) y Certified ScrumMaster

Socio fundador de Autentia (Desarrollo de software, Consultoría, Formación)

<mailto:alejandropg@autentia.com>

Autentia Real Business Solutions S.L. - "Soporte a Desarrollo"

<http://www.autentia.com>

A continuación puedes evaluarlo:

[Regístrate para evaluarlo](#)

Por favor, vota +1 o compártelo si te pareció interesante

Share |

Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

Puedes opinar o comentar cualquier sugerencia que quieras comunicarnos sobre este tutorial; con tu ayuda, podemos ofrecerte un mejor servicio.

(Sólo para usuarios registrados)

» **Regístrate** y accede a esta y otras ventajas «



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Impulsores

Comunidad

¿Ayuda?

0 personas han traído clicks a esta página

sin clicks



powered by [karmacray](#)

Copyright 2003-2012 © All Rights Reserved | [Texto legal y condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) |

