

# **Herramientas de calidad en código PHP**

**Por:  
@gabrielsaldana**

# **Los problemas mas comunes al programar en PHP**

# Los problemas mas comunes al programar en PHP

- código spaghetti
- malas prácticas
- no reutilizacion de código
- codigo complejo (anidaciones)
- falta de encapsulado (clases)
- limpieza de entradas/salidas
- falta de documentación
- mala consistencia en convenciones de código
- falta de optimización
- falta de pruebas

# Herramientas

# PHP Documentor



# PHP Documentor

Es una herramienta que te ayuda a generar la documentación del código de un proyecto automáticamente.

<http://phpdoc.org>

# PHP Documentor

```
/**
 * Counts the number of items in the provided array.
 *
 * @param mixed[] $array Array structure to count the elements of.
 *
 * @return int Returns the number of elements.
 */
function count(array $items)
{
    <...>
}
```

# PHP Documentor

The screenshot shows the phpDocumentor website interface. At the top, there is a purple header with the title 'phpDocumentor' on the right and navigation links 'INSTALL | README | ChangeLog | FAQ' on the left. Below the header, there are more navigation links: 'phpDocumentor | Converters | Opdf | HTML\_TreeMenu | Smarty | XML\_Beautifier'. The main content area is divided into a left sidebar and a main content pane. The sidebar contains a 'phpDocumentor' section with a 'Description' (Class trees, Index of elements, Todo List) and 'Tutorials/Manuals' (Package-level, phpDocumentor Guide to Creating Fantastic Documentation, phpDocumentor Quickstart, Source code for samples.php, Source code for sample2.php, Source code for sample3.php, phpDocumentor Tutorial, Documentable PHP Elements, phpDocumentor Tutorial, phpDocumentor Manual, phpDocumentor tags like @abstract, @access, @author, @category, @copyright, @deprecated, @example, @final, etc.). The main content pane has 'Previous' and 'Next' links at the top, with 'Next' pointing to 'phpDocumentor Quickstart'. The main heading is 'phpDocumentor Guide to Creating Fantastic Documentation'. Below the heading is a quote: 'What makes good documentation? This is unanswerable, but there are a few things to keep in mind'. The author is 'Gregory Beaver' with the email 'cellog@php.net'. A purple box highlights the section 'Why write good documentation for open source code?'. The text below this box states: 'Writing good documentation is essential to the success of any software project. The quality of documentation can be even more important than the quality of the code itself, as a good first impression will prompt developers to look further into your code. phpDocumentor is designed to make it easier to create documentation. phpDocumentor even makes it possible to generate separate sets of documentation from the same source!'

phpDocumentor

INSTALL | README | ChangeLog | FAQ

phpDocumentor | Converters | Opdf | HTML\_TreeMenu | Smarty | XML\_Beautifier

Previous

Next  
phpDocumentor Quickstart

## phpDocumentor Guide to Creating Fantastic Documentation

*What makes good documentation? This is unanswerable, but there are a few things to keep in mind*

**Gregory Beaver**  
cellog@php.net

### Why write good documentation for open source code?

Writing good documentation is essential to the success of any software project. The quality of documentation can be even more important than the quality of the code itself, as a good first impression will prompt developers to look further into your code. phpDocumentor is designed to make it easier to create documentation. phpDocumentor even makes it possible to generate separate sets of documentation from the same source!



# PHP Code Sniffer

- Es una herramienta que te revisa la sintaxis de tu código basado en
- algún estándar, ya sea alguno conocido como el de PEAR o el de Zend,
- o alguno definido por el usuario.
- 
- [http://pear.php.net/package/PHP\\_CodeSniffer/](http://pear.php.net/package/PHP_CodeSniffer/)

# PHP Code Sniffer

```
<?php
/* Hello.php */
function hello() {
    return "Hello World!";
}

echo hello();

?>
```

# PHPCS on a file

```
FILE: c:\localhost\test\hello.php
```

```
FOUND 3 ERROR(S) AND 0 WARNING(S) AFFECTING 2 LINE(S)
```

```
2 | ERROR | Missing file doc comment  
3 | ERROR | Missing function doc comment  
3 | ERROR | Opening brace should be on a new line
```

# PHPCS on a directory

```
c:\localhost\test>phpcs --report=summary ./ds
```

## PHP CODE SNIFFER REPORT SUMMARY

FILE	ERRORS	WARNINGS
c:\localhost\test\ds\link_list.php	8	4
c:\localhost\test\ds\linklist.class.php	64	30
c:\localhost\test\ds\LinkListTest.php	7	0
c:\localhost\test\ds\hello.php	3	0

A TOTAL OF 82 ERROR(S) AND 34 WARNING(S) WERE FOUND IN 4 FILE(S)

# PHP Depend



# PHP Depend

Analiza el código en busca de problemas y optimizaciones potenciales

# PHP Depend

- si tu clase o función es muy grande
- si tu función tiene demasiados parámetros
- nombres de variables muy cortos o muy largos
- demasiados ciclos anidados
- uso de eval()
- convenciones de nombres
- variables o métodos sin usar

# PHP Depend

<http://pdepend.org/>



# PHP Mess Detector

PHP  
*Mess Detector*

A stack of coins with a single coin on top, positioned to the right of the text 'PHP Mess Detector'. The stack consists of several coins with horizontal ridges, and the top coin is a plain, light-colored disc. The stack is rendered with a slight shadow and perspective.

# PHPMD

Un derivado de PHP Depend con una interfaz de configuración mas amigable.

<http://phpmd.org/>

# PHPCPD (Copy/Paste Detector)

Detecta líneas de código  
duplicadas

[https://github.com/sebastianbergmann/  
phpcpd](https://github.com/sebastianbergmann/phpcpd)

# PHPCPD

```
→ ~ phpcpd /usr/local/src/phpunit/PHPUnit  
phpcpd 1.4.0 by Sebastian Bergmann.
```

```
Found 3 exact clones with 53 duplicated lines in 5 files:
```

- /usr/local/src/phpunit/PHPUnit/Framework/Constraint/Or.php:136-157  
 /usr/local/src/phpunit/PHPUnit/Framework/Constraint/And.php:143-164
- /usr/local/src/phpunit/PHPUnit/Framework/Constraint/Or.php:136-157  
 /usr/local/src/phpunit/PHPUnit/Framework/Constraint/Xor.php:141-162
- /usr/local/src/phpunit/PHPUnit/Framework/Comparator/Scalar.php:121-132  
 /usr/local/src/phpunit/PHPUnit/Framework/Comparator/Numeric.php:102-113

```
0.19% duplicated lines out of 27640 total lines of code.
```

```
Time: 0 seconds, Memory: 18.25Mb
```

# XDebug




# XDebug

Es una extensión de PHP que provee capacidades de debugging y profiling. Puede mostrarte el stack, los parámetros y las funciones donde sucedió algún error. Puede mostrar cuanta memoria fue ocupada por el script y puede protegerte de recursiones infinitas, entre otras cosas más.

<http://xdebug.org>

# Xdebug

 Notice: Undefined variable: m in C:\www\local_vars.php on line 28				
Call Stack				
#	Time	Memory	Function	Location
1	0.0091	68064	{main}()	..\local_vars.php:0
2	0.0092	68736	foo( \$a = 1, \$b = 2 )	..\local_vars.php:11
3	0.0092	68960	bar( \$x = 4, \$y = 1 )	..\local_vars.php:15
4	0.0092	69264	baz( \$i = 7, \$k = 5 )	..\local_vars.php:20
Dump \$ _SERVER				
\$ _SERVER['HTTP_HOST'] =		string 'localhost' (length=9)		
\$ _SERVER['SERVER_NAME'] =		string 'localhost' (length=9)		
Variables in local scope (#4)				
\$m	Undefined			
	\$i =	int 7		
	\$k =	int 5		
	\$str =	string 'Hello World' (length=11)		
	\$t =	int 3		

# PHP Unit y Simpletest





# PHPUnit y Simpletest

Son frameworks para hacer pruebas unitarias (unit testing) y poder llevar a cabo el Test driven development.

<http://simpletest.org>

<http://phpunit.de>

# PHPUnit

## Unit Test Results

Designed for use with [PHPUnit](#) and [Zend Studio](#).

### Summary

Tests	Failures	Errors	Success rate	Time
4	1	0	75.00%	0.066

Note: *failures* are anticipated and checked for with assertions while *errors* are unanticipated.

### Overview

Name	Tests	Failures	Errors	Time(s)
<a href="#">CalculatorTest</a>	4	1	0	0.066

### TestCase CalculatorTest

Name	Status	Details	Time(s)
testAdd	Success		0.012
testDivide	Failure	PHPUnit_Framework_ExpectationFailedException: Failed asserting that <integer:3> is equal to <double:0.5>. Trace: PHPUnit_Framework_Assert::assertEquals() C:\Documents and Settings\shachar\workspace\Calculator 2\CalculatorTest.php:69	0.031
testMultiply	Error	PHPUnit_Framework_IncompleteTestError: multiply test not implemented Trace: PHPUnit_Framework_Assert::markTestIncomplete() C:\Documents and Settings\shachar\workspace\Calculator 2\CalculatorTest.php:79	0.013
testSubtract	Success		0.011

[Back to top](#)

Report generated at 2007-11-13T10:58:14+02:00

# Simpletest

Here's what the result of your first test would look like :

## My first test with SimpleTest

1/1 test cases complete: 1 passes, 0 fails and 0 exceptions.

Well not quite. In true TDD fashion, you should see a failing test case :

## My first test with SimpleTest

Fail: /Users/perrick/Sites/simpletest/test/array\_reporter\_test.php -> TestOfArrayReporter ->  
testContentOfArrayReporterWithOnePassAndOneFailure -> Equal expectation fails because [Integer: 2] differs from [Integer: 3] by 1 at  
[Users/perrick/Sites/simpletest/test/array\_reporter\_test.php line 13]

1/1 test cases complete: 0 passes, 1 fails and 0 exceptions.

**Preguntas?**

# Gracias

Gabriel Saldaña

[gabriel@gabrielsaldana.org](mailto:gabriel@gabrielsaldana.org)

Twitter/Identi.ca: @gabrielsaldana

Blog:

<http://blog.gabrielsaldana.org>