Hace poco anunciábamos el lanzamiento de PHP 5.4.0 y hablábamos de pasada sobre las nuevas características que incluye. Pues bien, hoy vamos a entrar en
mayor detalle
sobre los cambios de la nueva versión para comprobar de que forma
nos afecta
a la hora de desarrollar en
PHP
.

Uno de los cambios que más pueden afectar a nuestro código y nuestras aplicaciones es que el *"Safe Mode"* y *"Magic Quotes"* **han desaparecido** del lenguaje.

También se ha eliminado la posibilidad de **pasar argumentos por referencia** al invocar un método o función, es decir, no se puede llamar a una función utilizando una referencia:

```
// Esto resulta en un error fatal:
// Fatal error: Call-time pass-by-reference has been removed
$euros = 1000;
multiplica_estos(&$euros);
```

Si necesitamos pasar un valor por referencia a la función multiplicame es necesario hacerlo **en la declaración**
de la función:

```
function multiplicame(&$cantidad) {
...
}
```

Break y continue **ya no** aceptan argumentos variables aunque los argumentos estáticos siguen funcionando:

```
// Resulta en un error fatal
// PHP Fatal error:  'break' operator with non-constant operand is no longer supported
...
case 1:      echo "bla";      break 1 + 2;
...
```

A partir de esta versión **no se puede fijar** el *timezone* utilizando la variable de entorno TZ se obliga a fijar el valor de la opción date.timezone en
php.ini
o bien utilizar la funcióndate_default_timezone_set().

A su vez, **desaparecen** en esta versión las siguientes funciones:

- define_syslog_variables
- import_request_variables
- session_is_registered
- session_register
- session_unregister
- mysqli_bind_param
- mysqli_bind_result
- mysqli_client_encoding
- mysqli_fetch
- mysqli_param_count
- ,ysqli_get_metadata
- mysqli_send_long_data
- mysqli::client_encoding
- mysql_smtp::smtp

Novedades y adiciones al lenguaje

La versión 5.4.0 incorpora, obviamente, novedades al lenguaje, algunas de ellas muy esperadas relacionadas con los arrays. Por ejemplo ahora se pueden definir arrays con **sintaxis abreviada**
:

```
$page = [‘name’ => ‘Genbetadev’, ‘url’ => ‘http://www.genbetadev.com’, ‘category’ => ‘Technology’, ‘format’ => ‘Blog’];
```

Otra adición esperada es la **desreferencia de arrays** como valores de retorno de métodos y funciones, feature que incorporan otros lenguajes dinámicos como Python o JavaScript y en PHP
la verdad es que se
echaba de menos
:

```
...
$url = $page->getData()["url"];  // Amazing!!
```

Ahora podemos acceder a los miembros de una clase **al instanciar objetos** de la misma:

```
$page = (new Page($pageData))->initialize();
```

Otra adición que puede ser considerada muy potente o muy fea (y peligrosa) es la nueva sintaxis con posibilidad de **expresiones** para la llamada de métodos estáticos de las clases:

```
...
Page::{$url == 'http://www.genbetadev.com' ? 'vote' : 'novote' }();
...
```

A mi la verdad es que se me ocurren **infinitas posibilidades**.

Traits

Esta nueva versión del lenguaje incorpora un nuevo **mecanismo de reutilización** de código utilizado en lenguajes que no permiten la herencia múltiple como es el caso de
PHP
. Este nuevo mecanismo son los
traits
.

No debemos pensar sobre los traits en términos similares a una interfaz en Java. Al igual que las interfaces, un trait es similar a una clase pero solo define **agrupaciones de funcionalidad** mientras que una interfaz define que es lo que una clase
debe implementar
para cumplir con un contrato especificado.

No es posible instanciar un trait, al igual que no es posible instanciar una clase virtual pura (o
ADT
) en C++ o un interface en Java. Para utilizar un trait en nuestra clase utilizamos la palabra reservada use:

```
< ?php
  trait motorFunction {       function walk() { echo “I’m walking!”; }
}
class Animal {       use motorFunction;       function breath() { echo “I’m breathing!”; }
}
$cat = new Animal();
$cat->breath();
$cat->walk();
```

Con resutado:

```
I'm breathing!
I',m walking!
```

El uso de traits es **bastante complejo** y es, creo, la mayor adición al lenguaje en esta nueva versión.

La precedencia de métodos heredados de una clase base puede ser sobreescrita por un método miembro en un trait que adopta además el papel de hija de la clase padre inicial al usarla en una clase hija:

```
< ?php
class Animal {	public function walk() {		echo "an animal walks";	}
}
trait slowlyWalk {	public function walk() {		parent::walk();		echo " slowly";	}
}
class Cat extends Animal {	use slowlyWalk;
}
$cat = new Cat();
$cat->walk();
```

Que produce el siguiente resultado:

```
an animal walks slowly
```

Recomiendo que le echéis un vistazo a la página sobre traits en el manual de ayuda de PHP pues hay
muchísimos más casos de uso
que no hemos tratado aquí hoy como la herencia de múltiples traits o la resolución de

conflictos.

CLI Web Server

Por último, pero no por ello menos importante, cabe destacar que esta nueva versión de PHP viene con un s
ervidor web
para pruebas de desarrollo embebido (y que no debe usarse jamás para producción) y su uso es bastante sencillo:

```
$ cd ~/MyProject
$ php -S localhost:8000
PHP 5.4.0--pl0-gentoo Development Server started at Tue Mar 20 12:51:09 2012
Listening on localhost:8000
Document root is /home/damnwidget/MyProject

Press Ctrl-C to quit.
```

De momento esto es todo, como podéis comprobar, la nueva versión de PHP viene con **novedades y cambios muy esperados**
y no dudamos en que el futuro, el lenguaje seguirá mejorando.

Más informacón | Página web de PHP

Fuente: http://www.genbetadev.com/php/php-5-4-a-fondo