

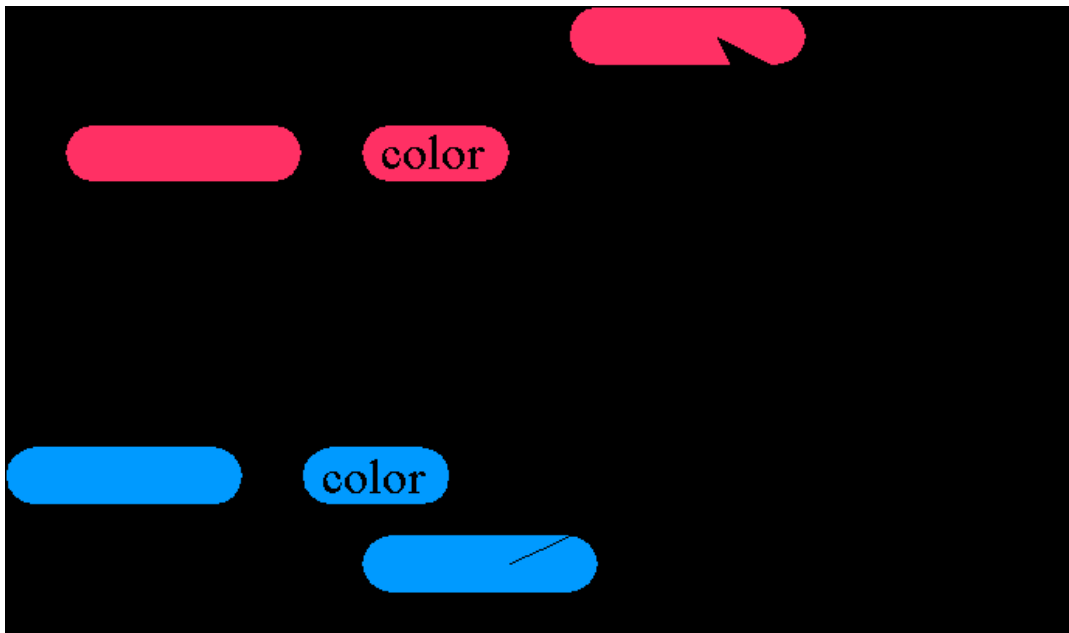
# Migrating from PHP 4 to PHP 5

Intl. PHP Conference

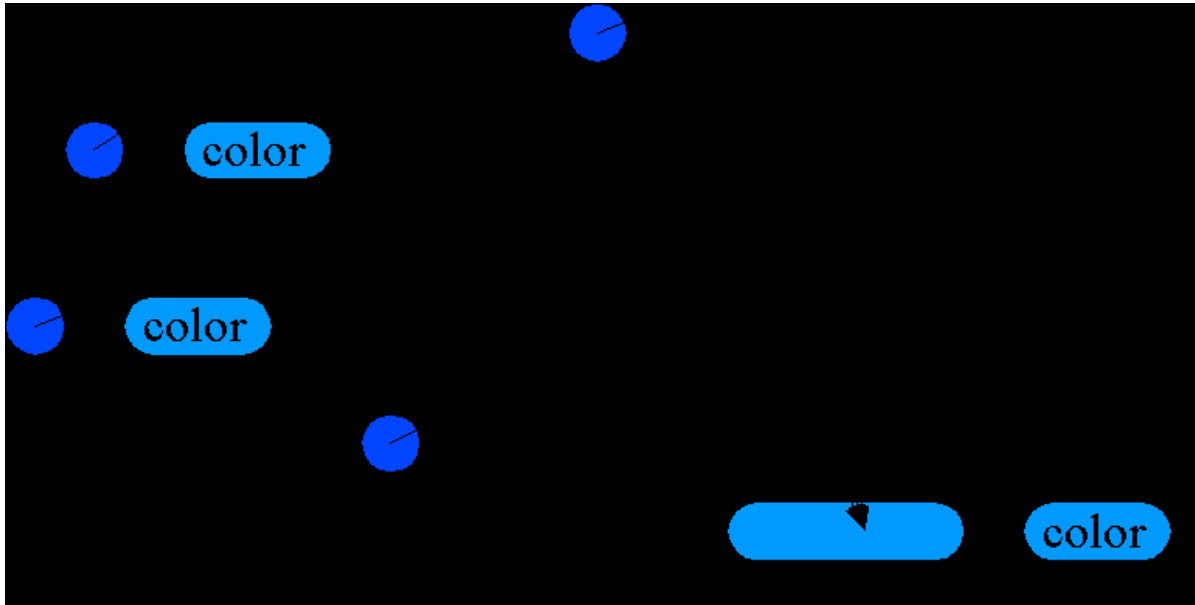
May 5, 2004. Amsterdam, the Netherlands

Derick Rethans <derick@php.net>

<http://pres.derickrethans.nl/migrating-ffmpeg>



- o Pass by value
- o Use of references needed almost everywhere...  
at call time, and even during instantiation.



- o Pass by reference
- o Objects are referenced by a handle.

### The PHP 5 Way

```
<?php
class OS {
    var $name;

    function OS($name) {
        $this->name = $name;
    }
}

function changeName($obj, $name) {
    $obj->name = $name;
}

$linux = new OS('linux');
$win = clone $linux;
changeName($win, 'windows');
echo $linux->name. "\n";
echo $win->name;

? >
```

### Output:

```
linux
windows
```

## The Compatible Way

```
<?php
    ini_set('zend.zel_compatibility_mode', '1');

    class OS {
        var $name;

        function OS($name) {
            $this->name = $name;
        }
    }

    function changeName(&$obj, $name) {
        $obj->name = $name;
    }

    $linux = new OS('linux');
    $win = $linux;
    changeName($win, 'windows');
    echo $linux->name. "\n";
    echo $win->name;

? >
```

### Output:

```
linux
windows
```

## PHP 4

```
<?php
class Country {
    function set_name($name) {
        $this->name = $name;
    }
}

$jurop = new Country;
$dutchieland = new Country;
$dutchieland->set_name('The Netherlands');

echo (int) $jurop, "\n";
echo (int) $dutchieland, "\n";
? >
```

## Output:

```
0
1
```

- o In PHP 4 (int) \$object results in:
- o "0" when there are no properties set
- o "1" when there are properties set

## PHP 5

```
<?php
class Country {
    function set_name($name) {
        $this->name = $name;
    }
}

$jurop = new Country;
$dutchieland = new Country;
$dutchieland->set_name('The Netherlands');

echo (int) $jurop, "\n";
echo (int) $dutchieland, "\n";
? >
```

## Output:

```
Notice: Object of class Country could not be converted to integer in - on line
13
1
Notice: Object of class Country could not be converted to integer in - on line
14
1
```

- o In PHP 5 (int) \$object results in:
- o A notice and always "1"
- o unless zend.ze1\_compatibility\_mode = 1

## PHP 4

```
<?php
class Country {
    var $area;
    function Country($area) {
        $this->area = $area;
    }
}

$deutschland = new Country('West Europe');
$die_schweiz = new Country('West Europe');
$norway      = new Country('Scandinavia');

if ($deutschland == $die_schweiz) {
    echo "Deutschland ist die Schweiz.\n";
}
if ($deutschland == $norway) {
    echo "Deutschland er Norway.\n";
}
? >
```

- o In PHP 4 `$obj1 == $obj2` results in true:
- o when the object's properties are the same

## PHP 5

```
<?php
class Country {
    var $area;
    function Country($area) {
        $this->area = $area;
    }
}

$deutschland = new Country('West Europe');
$die_schweiz = new Country('West Europe');
$norway      = new Country('Scandinavia');

if ($deutschland == $die_schweiz) {
    echo "Deutschland ist die Schweiz.\n";
}
if ($deutschland == $norway) {
    echo "Deutschland er Norway.\n";
}
? >
```

- o In PHP 5 `$obj1 == $obj2` results in true:
- o when the object's handles are the same

php.ini:

```
zend.ze1_compatibility_mode = 1
```

Affects:

- o cloning objects
- o casting objects
- o comparing objects

# Assigning to \$this

---

PEAR uses this for two things mainly:

1. returning errors from constructors (`$this = PEAR_Error();`)
2. driver model (`$this = new Slider('Pager');`)

This is no longer allowed in PHP 5, so the classes should be redesigned.

Solutions (not compatible with PHP 4).

1. Use Exceptions
2. driver model (`$this = new Slider('Pager');`)

```
<?php
class TheNetherlands {
    var $area;
    function TheNetherlands($area) {
        $this->area = $area;
    }
}

$holland = new TheNetherlands('Holland');

echo get_class($holland), "\n";
? >
```

PHP 4:

thenetherlands

PHP 5:

TheNetherlands

E\_STRICT is a new error level.

- o Not part of E\_ALL
- o Numerical value is 2048 (and E\_ALL is still 2047)
- o To see all errors: E\_ALL | E\_STRICT

Affects:

- o Automagically creating objects
- o var / public modifiers
- o Constructor naming

```
<?php
class Russia {
    var $local_name;

    function Russia() {
        $this->local_name = "Roosjia";
    }
}
? >
```

**PHP 5:**

Strict Standards: var: Deprecated. Please use the public/private/protected modifiers in - on line 3

```
<?php
    $country->name = 'Moldovia';

    var_dump($country);
? >
```

**PHP 5:**

```
Strict Standards: Creating default object from empty value in - on line 2
object(stdClass)#1 (1) {
    ["name"]=>
        string(8) "Moldovia"
}
```

```
<?php
class Scandinavia {
    function get_countries() {
        return array('Norway', 'Sweden', 'Denmark', 'Finland');
    }
}

class Europe extends Scandinavia {
    function get_countries($area) {
        if ($area == "Scandinavia") {
            return Scandinavia::get_countries();
        }
    }
}
? >
```

**PHP 5:**

Strict Standards: Declaration of Europe::get\_countries() should be compatible with that of Scandinavia::get\_countries() in - on line 14

In the PHP 4 distributions you will find the following binaries:

- o php.exe (The CGI binary to use with a web server)
- o cli/php.exe (The CLI binary to use with command line scripts)

In the PHP 5 distributions you will find the following binaries:

- o php-cgi.exe (The CGI binary to use with a web server)
- o php.exe (The CLI binary to use with command line scripts)

In PHP 5 the MySQL client library is no longer bundled because:

- o Maintainability
- o License change
- o Problems with other modules

Solution:

- o Install the MySQL client libraries
- o `--with-mysql=[DIR]`

# array\_merge()

---

array\_merge() no longer accepts non-array parameters to merge.

```
<?php
    $scandinavia = array('Denmark', 'Norway', 'Sweden', 'Finland');
    $uk = "The UK";

    $europe = array_merge($scandinavia, $uk);

    print_r($europe);
? >
```

PHP 4:

```
Array
(
    [0] => Denmark
    [1] => Norway
    [2] => Sweden
    [3] => Finland
    [4] => The UK
)
```

PHP 5:

```
Warning: array_merge(): Argument #2 is not an array in - on line 7
```

# strrpos() and stripos()

---

str(i)pos() now searches the full needle inside the haystack, and not only the first character of it.

```
<?php
    $haystack = "There are now ten more countries in Europe";
    $needle   = "The Netherlands";

    $pos = strrpos($haystack, $needle);
    var_dump($pos);
? <
```

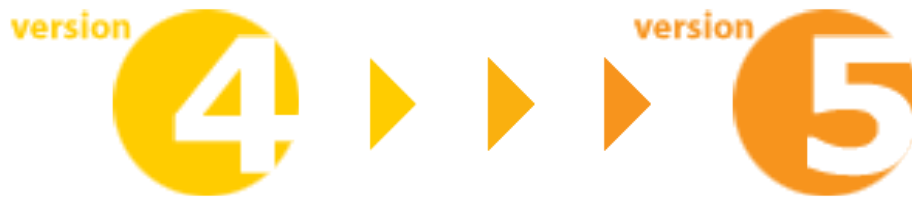
PHP 4:

```
int(0)
```

PHP 5:

```
bool(false)
```

?



These Slides: <http://pres.derickrethans.nl>

PHP: <http://www.php.net>

PHP 4 to 5 Migration manual page: <http://www.php.net/manual/en/migration5.php>

Questions?: [derick@php.net](mailto:derick@php.net)

# Index

Objectmodel in PHP 4 .....	2
Objectmodel in PHP 5 .....	3
Clone .....	4
Clone .....	5
Casting Objects .....	6
Casting Objects .....	7
Comparing Objects .....	8
Comparing Objects .....	9
Compatibility Mode .....	10
Assigning to \$this .....	11
get_class() .....	12
E_STRICT .....	13
var vs. public .....	14
Automagic Objects .....	15
Signatures .....	16
Command Line Interface .....	17
Changes in MySQL .....	18
array_merge() .....	19
strrpos() and stripos() .....	20
Questions .....	21
Resources .....	22