

Introduction to PHP 5.2.1

Ilia Alshanetsky
PHP Quebec Conference
Quebec, Canada

Why Migrate?

Because PHP 4

Why Migrate?

Because PHP 4

SUCKS!!!!

Why Migrate?

— [**New Features**

— [**Improved Performance**

— [**Much Better Security**

— [**Greater Stability**

— [**Actively supported**

New Features in PHP 5.2

— [New Extensions

- JSON (JavaScript Object Notation)
- Filter Extension (simple input validation)
- ZIP (Full zip compression support read & write)
- Date (date manipulation functions/objects)

New Features in PHP 5.2

— [`__toString()` now works everywhere

— [`E_RECOVERABLE_ERROR` (fewer fatal errors)

— [New SPL features (Regex Iterators, SplFileObject CSV support, Caching Iterator)

— [Data: stream support

— [And many other “minor” features.

Performance Enhancements

- [New & Improved Memory Manager + Heap Protection
- [Faster include/require_once
- [Optimized str_replace() and implode() functions
- [Faster try {} catch {} blocks
- [Significantly faster performance on Win32
- [Optimized shutdown sequence
- [Many other, less memorable optimizations

Improved Security

- [New configuration option **allow_url_include** (disabled by default)
- [Over 40 security fixes compared to any prior release.
- [More accurate memory usage tracking
- [Filter extension can help filter out hostile input preventing XSS, SQL Injection and other nastiness.
- [Memory limit is now always enabled (128M default limit).

Improved Stability

PHP 5.2.1 includes hundreds of bug fixes compared to previous releases, over **300** since 5.1.6



Chances are that if you've reported a PHP bug in the last 6-8 months, PHP 5.2.1 has the fix for it.

Brief guide to PHP Support ;-)

PHP 4.X branch - Discontinued (EOL)

PHP 5.0 branch - Abandoned

PHP 5.1 branch - Abandoned

PHP 5.2 - Actively developed, rapid bug fixes

average lifetime of a bug is about 4 days

PHP 6 - Ω°C ✂ ☂ ✨ β: 🍷



Migrating from PHP 5.0/5.1

— [The value of **E_ALL** has changed, now includes **E_RECOVERABLE_ERROR**.

— If you don't change error reporting level you won't see recoverable errors displayed or logged

— You may need to adjust `error_handler()` function.

— [If you are loading PHP files via http/ftp via `include()` or `require` you will need to enable **`allow_url_include`**.

— [No more **abstract static classes**.

— [2 new native classes **DateTime** and **DateTimeZone**, you'll need to rename yours they carry the same names.

— [To ensure proper time-zone settings and avoid `date()` and alike functions from generating error messages you need to set the **date.timezone** INI setting. (Since PHP 5.1.0)

— [**FilePro**, **Informix** and **hwapi** extensions are gone (moved to PECL).

— [You'll probably need to increase **memory_limit**.

— [In CLI PHP no longer checks for php.ini or php-cli.ini inside the PWD (current directory).

— [On Win32 PHPRC environment variable has priority over the registry for php.ini searching.

— [When converted to strings, objects without `__toString()` no longer return object id.

— `<5.2 (string)new stdClass == Object id #1`

— `5.2> (string)new stdClass == E_RECOVERABLE_ERROR`

— [SQLite library in pdo_sqlite was upgraded to 3.3.7

— 5.2.2 will have version 3.3.13

— [Fortunately **it is backward compatible** to earlier 3.2.8 version, so there are no issues.

— [The SQLite extension still offers access to sqlite2 databases.

PHP 5.0 specific changes

Inheritance overloading rules are a little bit stricter.

```
class a {  
    function &test () {  
        $foo = "val";  
        return $foo;  
    }  
}
```

```
class b extends a {  
    function test () {  
        return "val";  
    }  
}
```

In PHP 5.2. this will emit **E_STRICT** warning message

```
class foo {  
  const foo = "bar";  
  const foo = "baz";  
}
```

foo::foo == "baz"

```
class a {  
  const bar = "foo";  
}  
class b extends a {  
  const bar = "zoom";  
}
```

b::bar == "zoom"

```
class foo {  
  const foo = "bar",  
  const foo = "baz";  
}
```

FATAL ERROR
foo::foo == "baz"

```
class a {  
  const bar = "foo";  
}  
class b extends a {  
  const bar = "zoom";  
}
```

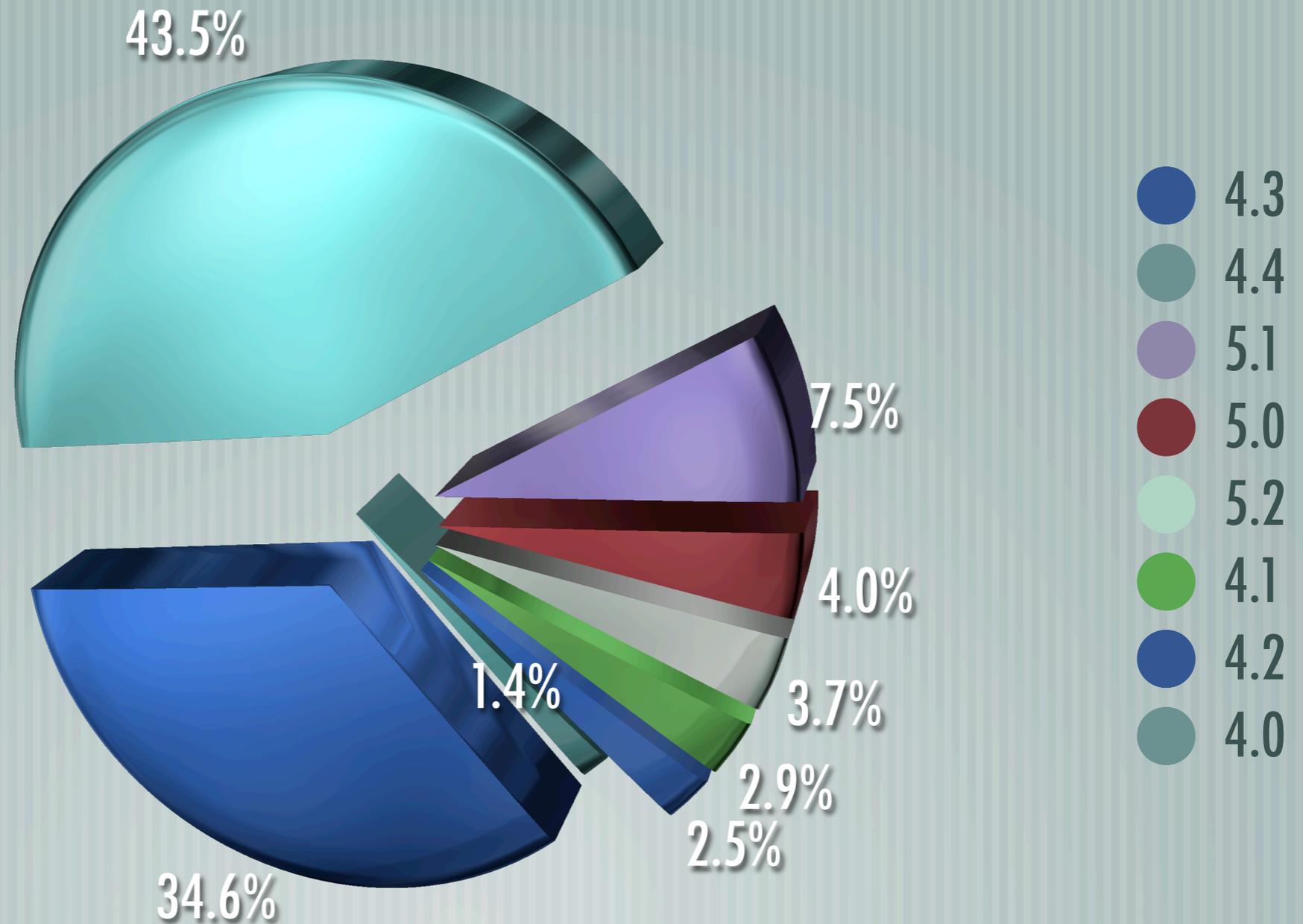
b::bar == "zoom"

**Bottom Line: Upgrading
from 5.X to 5.2 is easy!**

From PHP 4 to PHP 5.2

PHP 4 is still most popular

PHP 5 currently holds **15%** of the market.



How to migrate from 4 to 5.2

— [Major differences

- Object handling (by-ref vs by-value)
- DomXML to DOM extension change
- E_STRICT error mode
- MySQL extension no longer includes the MySQL library

Objects are finally by-ref!

— [Arguably the biggest change in PHP 5 compared to PHP 4 is in the way objects are handled.

— [In PHP 5 they are finally being passed by reference.

— [Fortunately this change breaks very few applications.

```
<?php
```

PHP 4

```
$a =& new Class;
```

```
function foo(&$obj) {  
    $obj->prop = 123;  
}
```

```
$obj_copy = $obj;
```

```
function &a() {  
    $a =& new Object;  
    return $a;  
}
```

```
<?php
```

PHP 5.2

```
$a = new Class;
```

```
function foo($obj) {  
    $obj->prop = 123;  
}
```

```
$obj_copy = clone($obj);
```

```
function a()  
{  
    return new Object;  
}
```

```
<?php
```

PHP 4

```
$a =& new Class;
```

```
function foo(&$obj) {  
    $obj->prop = 123;  
}
```

```
$obj_copy = $obj;
```

```
function &a() {  
    $a =& new Object;  
    return $a;  
}
```

```
<?php
```

PHP 5.2

```
$a = new Class;
```

```
function foo($obj) {  
    $obj->prop = 123;  
}
```

```
$obj_copy = clone($obj);
```

```
function a()  
{  
    return new Object;  
}
```



needs
change

Object upgrade myths

— [**Fiction:** **var** keyword generates warnings

— **Fact:** Since PHP 5.1.6, it is silently translated to **public**.

— [**Fiction:** old style constructors won't work

— **Fact:** Old style constructors still work perfectly.

— [**Fiction:** code will break if I don't remove &

— **Fact:** completely unnecessary

— [**Fiction:** PHP is turning into Java, and **I hate Java**

— **Fact:** As long Marcus is not in charge ;-)

DOMXML Replaced By DOM

— [If you've used DOMXML extension to parse XML, well, it is time to rewrite your code.

— New extension is faster

— Does not leak memory

— It works, as opposed to; on good days, unless they fall on a Friday.

— [However, it can still be installed from PECL repository.

New Error Mode, E_STRICT

— [Designed to identify deprecated behavior & functionality

— [Will new errors be spewed to screen or filling my log files?

— **No**, E_STRICT error reporting needs to be explicitly enabled.

— [Do I need to change my configuration file?

— **Not at all.**

— [Do I need to change/rewrite my code?

— **Not until PHP 6**, and even then...

A little more strict

```
$a = "test";  
$a[0][1][2] = "a";
```

In PHP 4 this was a warning.

```
array_merge("string", 123);
```

In PHP 4 this **worked**.

```
class foo {  
    function a($obj) { $this = $obj; }  
}
```

A little more strict

```
$a = "test";
```

```
Fatal Error  
$a[0][1][2] = "a";
```

In PHP 4 this was a warning.

```
array_merge("string", 123);
```

In PHP 4 this **worked**.

```
class foo {  
    function a($obj) { $this = $obj; }  
}
```

A little more strict

```
$a = "test";
```

```
$a[0][1][2] = "a";
```

Fatal Error

In PHP 4 this was a warning.

```
array_merge("string", 123);
```

Warning & NULL

In PHP 4 this **worked**.

```
class foo {  
    function a($obj) { $this = $obj; }  
}
```

A little more strict

```
$a = "test";
```

```
$a[0][1][2] = "a";
```

Fatal Error

In PHP 4 this was a warning.

```
array_merge("string", 123);
```

Warning & NULL

In PHP 4 this **worked**.

```
class foo {
```

```
function a($obj) { $this = $obj; }
```

```
}
```

Fatal Error

A few minor behavior changes

`strrpos()` searches for the complete "needle" rather than just the 1st character.

```
<?php  
var_dump(strrpos("kangaroo", "an"));  
// int(4)
```

PHP 4

PHP 5.2

```
<?php  
var_dump(strrpos("kangaroo", "an"));  
// int(1)
```

```
class Foo {      PHP 4
    function AbC() {}
}
class baR extends foo {}
```

```
$a = new bar;
echo get_class($a);
// bar
print_r(
    get_class_methods($a)
); // array("abc");
echo get_parent_class($a);
// foo
```

```
class Foo {      PHP 5.2
    function AbC() {}
}
class baR extends foo {}
```

```
$a = new bar;
echo get_class($a);
// baR
print_r(
    get_class_methods($a)
); // array("AbC");
echo get_parent_class($a);
// Foo
```

```
var_dump(  
ip2long("000.000.999.888"),  
ip2long("255.255.255.255")  
);  
// int(-1)  
// int(-1)
```

```
$a = new stdClass;  
var_dump(empty($a));  
// bool(true)
```

```
var_dump(  
ip2long("000.000.999.888"),  
ip2long("255.255.255.255")  
);  
// bool(false)  
// int(-1)
```

```
$a = new stdClass;  
var_dump(empty($a));  
// bool(false)
```

Reference Fixes (4.4 has this too)

When a function expects a reference and is given a non-reference-able value, `E_STRICT` will be raised and parameter will be passed by value.

```
array_pop(explode("a", "abc"));
```

```
$a = 123;  
function foo() { return $GLOBALS['a']; }  
$b = &foo();
```

Deprecated Functionality

— [Things you should avoid using in the future:

— **is_a()** - replaced by **instanceof** operator

— `is_a($obj, "foo");`  `$obj instanceof foo`

— **dl()** - use **extension=** in php.ini

— `mktime()` without any parameters, just use `time()`

Deprecated Functionality

Daylight savings parameter for mktime()

— `mktime(h,m,s,m,d,y,dst);`

Calling non-static methods statically

— `class foo { function bar() {} } foo::bar();`

Using `{}` to access string offsets

— `$a = "foo"; $a{0} = "F";`

So long MySQL ... Hello SQLite

Well, not really.

As of PHP 5 MySQL client library is no longer bundled due to licensing issues.



All this means is that **mysql**/mysql/pdo_mysql extensions are **not enabled by default**.

Not really a concern unless you compile PHP yourself.

Migration Code Validator

Directory
to scan

PHP script extensions

```
find /source/directory \
-name \*.php -o -name \*.inc \
| xargs -n1 \
php -ddisplay_errors=1 -derror_reporting=8191 -l
```

E_ALL |
E_STRICT

Thank you for listening

Additional Resources

- These slides: <http://www.ilia.ws>
- PHP 4 to 5 migration docs: <http://php.net/manual/migration5.php>
- PHP 5.2 changes: <http://www.php.net/ChangeLog-5.php#5.2.1>
- Upgrade Guide http://www.php.net/UPDATE_5_2.txt