

PHP 5.4 arrive...





Hello, I'm Julien Pauli:)

- @julienpauli **
 - http://julien-pauli.developpez.com



- IRC Freenode, EFNet
 - @jpauli
- Software architect PHP Guru
- Working at Comuto
 - http://www.covoiturage.fr



- OSS Contributor
 - PHP Contributor http://doc.php.net/fr
 - jpauli@php.net



Orateur prévu : David Soria Parra

From: David Soria Parra

To: communication / AFUP

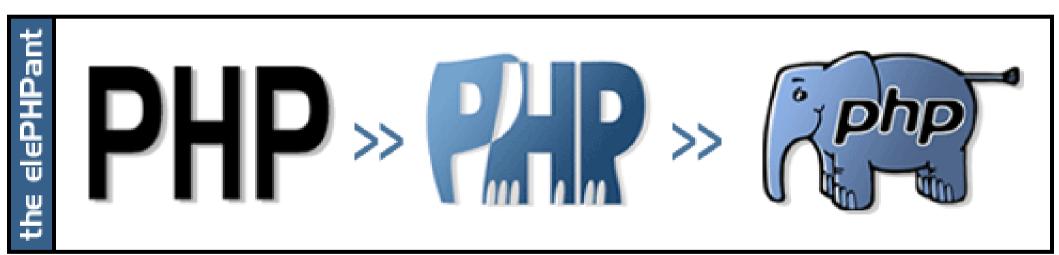
Hello,

I'm very very sorry, but I have to cancel my talk on Thursday. I'm sick and have to stay at home for the week. This is the first time I have to cancel a talk at all and I'm very very sorry for it.

[...]



On va parler de PHP





PHP & PHP 5.4?

- PHP : Une belle histoire (historique)
- Aujourd'hui : la communauté évolue
 - Un nouveau processus de contribution
 - De nouveaux canaux de développement
 - Une nouvelle timeline de distribution
- Nouveautés techniques
 - Performances
 - Fonctionnalités
- Conclusion



Historique

• 1995 : PHP 1.0

1998 : PHP 3.0

2000 : PHP 4.0

2004 : PHP 5.0

• 2009 : PHP 5.3

2011: PHP 5.4

PHP 6 ?

http://museum.php.net



Aujourd'hui:

"All PHP users should note that the PHP 5.2 series is NOT supported anymore."

http://www.php.net



La communauté évolue -Les RFC



RFC

- Une idée pour améliorer PHP ?
- Publier une RFC
- La réfléchir, la mûrir
- Publier des patchs, des squelettes
 - Les mûrir
- Faire voter la RFC
 - https://wiki.php.net/rfc/voting

"There's no way around this 'small' issue. Changes made to the PHP language will affect millions of people, and theoretically, each and every one of them should have a say in what we do. For obvious reasons, though, this isn't a practical approach."

https://wiki.php.net/rfc/voting



Requests For Comments

https://wiki.php.net/rfc

In voting phase

- strn(case)cmp supporting a negative length as its third paramter
 This extends PHP's string function strn(case)cmp to support a negative length as parameter
- SplClassLoader
 Merge implementation of first proposal of PHP Standards Recommendation (PSR) into SPL core

Under Discussion

- foreach_variable supporting T_LIST
 This extends PHP's language parser to support T_LIST in foreach_variable
- Enhanced Error Handling
 This offers a concept and implementation to let the user (php coder) choose between php_error, exception et al while retaining
- Parameter (and Return) value type hints/checks
 Several RFC that propose to complete the actual parameter type hints/checks (and an implementation for return value type hin
- Zend Signal Handling
 Improve stability and speed when running under any forking SAPI by adding deferred signal handling to the Zend Engine.
- Native TLS in ZTS
 Improving performance of ZTS builds by using native thread local storage.
- Better benchmarks for PHP
 A proposal for replacing current bench.php benchmarking script.
- Remove reliance of Zend API
 A proposal to remove the use of the Zend API in PHP libraries.
- PHP Native Interface
 Provide complete alternative to writing libraries interfaces using the Zend API libraries.



Nouveau processus de publication Release process



Nouveau processus de publication

"PHP releases have always been done spontaneously, in a somehow chaotic way. Individual(s) decided when a release will happen and what could or could fit in. Release managers role are unclear and the way to nominate them is not clearly defined either.

The goals of this RFC aim to solve these issues while giving to us, our users and 3rd parties (distributions, contributors, etc.) more visibility and the ability to actually have a roadmap, or plan developments."

https://wiki.php.net/rfc/releaseprocess



Un calendrier plus précis

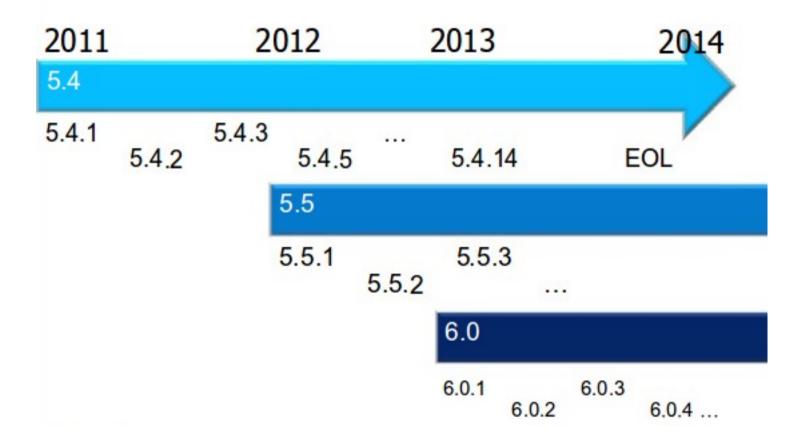
- Une durée de vie de 3 ans
 - Une mineure par an





Un calendrier compréhensif

- 2 majeures peuvent cohéxister
 - Comme PHP4 & PHP5 à leur époque
 - Simplifier les migrations





On ne casse pas tout

5.45.56.0

BC Break not allowed

BC Break allowed (if desired)

- Majeures (PHP 5)
 - BC breaks possibles
- Mineures (PHP 5.4)
 - nouvelles fonctionnalités
 - Mouvements d'extensions (PECL ↔ Core)
 - Pas de BC break
- Révisions (PHP 5.4.1)
 - Bug fixes seulement



Nouveau processus de publication

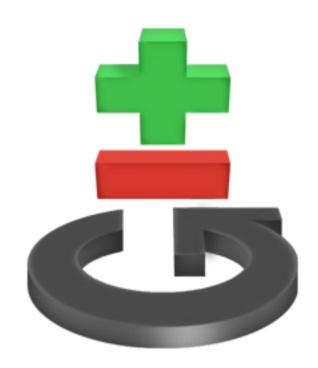
https://wiki.php.net/rfc/releaseprocess



Un nouveau processus de développement



Un nouveau processus de développement





PHP passe à Git

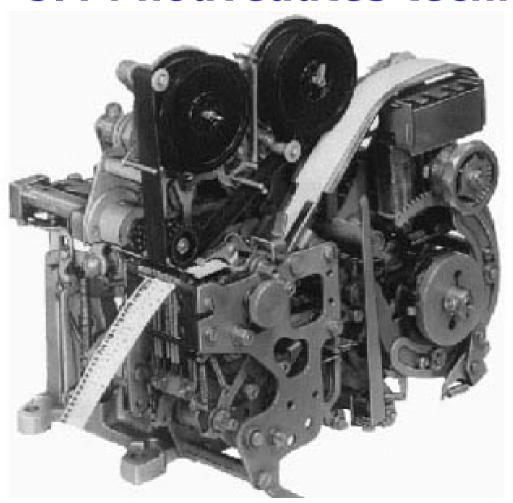
- 24/12/2011
 - Passage à Git
- Serveur git hebergé sur php.net
 - Pour des raisons de sécurité et de gestion des utilisateurs
- Mirroir sur github



Byebye SVN



PHP 5.4 : nouveautés techniques





PHP 5.4, un peu de ménage

- Dites au revoir à :
 - safe_mode
 - register_globals, import_request_variables()
 - register_long_arrays
 - allow_call_time_pass_reference (foo(&\$bar))
 - y2k_compliance
 - magic quotes***
 - session_register(), session_unregister(), session_is_registered()
 - ext/sqlite



PHP 5.4, ajouts en vrac

- new ReflectionZendExtension
- new SessionHandler
- Compatibilité avec autoconf 2.59+ (obligatoire)
- MySQLnd activé par défaut (et non plus libmysql)
- http_response_code()
- session status()
- ReflectionClass::newInstanceWithoutConstructor()
- Dtrace support (mac & Solaris)
- E_ALL contient E_STRICT ;-)



Le meilleur de PHP 5.4 ... ??



Performances!!

(~+55% bruts)



Performances de PHP5.4

- La plus belle amélioration de PHP 5.4 est sous le manteau
 - http://svn.php.net/viewvc/php/php-src/branches/PHP_5_4/NEWS
 - Improved Zend Engine, performance tweaks and optimizations: (Dmitry)
 - . Inlined most probable code-paths for arithmetic operations directly into executor.
 - . Eliminated unnecessary iterations during request startup/shutdown.
 - . Changed \$GLOBALS into a JIT autoglobal, so it's initialized only if used. (this may affect opcode caches!)
 - . Improved performance of @ (silence) operator.
 - . Simplified string offset reading. \$str[1][0] is now a legal construct.
 - . Added caches to eliminate repeatable run-time bindings of functions, classes, constants, methods and properties.
 - . Added concept of interned strings. All strings constants known at compile time are allocated in a single copy and never changed.
 - . Added an optimization which saves memory and emalloc/efree calls for empty HashTables. (Stas, Dmitry)

[...]

0.202

9.988



0.557

0.575

0.555

0.623

0.617

0.611

0.617

0.977

0.974

0.962

0.979

0.737

0.982

0.607

0.643

0.608

0.533

0.537

0.554

25 816

0.373

0.353

0.421

0.415

0.409

0.415

0.775

0.772

0.760

0.777

0.535

0.780

0.405

0.441

0.406

0.331

0.335

0.352

1.858

0.150

func()

undef func()

x = self::x

isset(self::\$x)

empty(self::\$x)

self::\$x = 0

x = Foo::x

isset(Foo::\$x)

empty(Foo::\$x)

 $x = \frac{1}{2}$

\$this->x += 2

this->x=0

++\$this->x

--\$this->x

\$this->x++

Total

Foo::\$x = 0

self::f()

Foo::f()

int func()

PHP 5.4, micro-bench func() under func

<--PHP 5.3 : 25.8 sec</p>

PHP 5.4 : 9.9sec →

Micro bench dispo sur svn

```
0.121
                   0.366
                             0.245
undef func()
                   0.366
                             0.246
int func()
                   0.324
                             0.204
x = self::x
                   0.268
                             0.147
self::$x = 0
                   0.263
                             0.142
isset(self::$x)
                   0.226
                             0.105
empty(self::$x)
                   0.235
                             0.115
x = Foo::x
                   0.231
                             0.111
Foo::$x = 0
                   0.318
                             0.197
isset(Foo::$x)
                   0.200
                             0.079
empty(Foo::$x)
                   0.214
                             0.093
self::f()
                   0.389
                             0.268
                             0.222
Foo::f()
                   0.343
$x = $this->x
                   0.241
                             0.120
this->x=0
                   0.305
                             0.184
this->x += 2
                             0.118
                   0.238
++$this->x
                   0.215
                             0.095
 -$this->x
                   0.310
                             0.190
$this->x++
                   0.237
                             0.116
$this->x--
                   0.244
                             0.123
isset($this->x)
                   0.223
                             0.102
empty($this->x)
                   0.241
                             0.121
$this->f()
                   0.394
                             0.274
$x = Foo::TEST
                   0.221
                             0.100
new Foo()
                   0.750
                             0.630
$x = TEST
                   0.176
                             0.055
$x = $GET
                   0.250
                             0.129
x = GLOBALS['v']
                   0.337
                             0.217
x = \frac{\sinh(v')}{v'}
                   0.248
                             0.127
x = str[0]
                   0.357
                             0.236
x = a ?: null
                   0.242
                             0.121
x = f ?: tmp
                   0.323
                             0.202
x = f ? f : a
                             0.128
                   0.249
```

\$x = \$f ? \$f : tmp 0.323

Total

\$this->x--0.556 0.354 isset(\$this->x) 0.596 0.394 empty(\$this->x) 0.614 0.412 0.702 0.500 \$this->f() x = Foo::TEST0.295 0.497 1.830 1.628 new Foo() \$x = TEST0.484 0.282 \$x = \$GET0.465 0.262 x = GLOBALS['v'] 0.7540.552 $x = \frac{hash['v']}{}$ 0.474 0.272 x = str[0]0.420 0.622 x = a ?: null1.988 1.786 x = f ?: tmp0.370 0.168

\$x = \$f ? \$f : \$a 2.060

\$x = \$f ? \$f : tmp 0.352



PHP 5.4, essai PHPUnit

- Lançons la suite de tests de Symfony2
 - Symfony2 master
 - PHPUnit 3.6.3

Time: 23 seconds, Memory: 294.50Mb

Time: 15 seconds, Memory: 136.50Mb



PHP 5.4 - sort open tags

<?= est toujours disponible quel que soit le niveau de sort_open_tags



Serveur web embarqué

- Pas pour de la production ;-)
 - Possibilité d'utiliser un routeur (script PHP)
- http://www.php.net/manual/fr/features.commandline.webserver.php

```
julien@jpauli:/usr/local/php54/bin$ ./php -S 127.0.0.1:1030 -t /home/julien/www
PHP 5.4.0RC1 Development Server started at Wed Nov 23 13:55:29 2011
Listening on 127.0.0.1:1030
Document root is /home/julien/www
Press Ctrl-C to quit.
[Wed Nov 23 13:55:40 2011] 127.0.0.1:45687 [200]: /phpinfo.php
```

PHP Version 5.4.0RC1



| | Linux jpauli 2.6.38-12-generic #51-Ubuntu SMP Wed Sep 28
14:27:32 UTC 2011 x86_64 |
|-------------------|--|
| Build Date | Nov 18 2011 10:24:21 |
| Configure Command | './configure' 'prefix=/usr/local/php54' 'with-config-file-scan-
dir=/usr/local/php54/etc' |
| Server API | Built-in HTTP server |



Callable

- Nouveau typage : "callable"
 - "callable" devient donc un mot reservé

```
class PubSub {
   // code..
   public function subscribe($name, callable $cb) {
       if (!isset($this->callbacks[$name])) {
          $this->callbacks[$name] = array();
       $this->callbacks[$name] = $cb;
$ps = new PubSub();
$ps->subscribe('foo', function() { echo 'test'; });
$ps->subscribe('foo', 'bla'); // catchable fatal error
```



Qu'est ce qui est "callable"?

- Toute callback PHP :
 - 'some-existing-function'
 - array(\$someobj, 'some-public-method')
 - array('someClass', 'some-static-public-method')
 - function () { };
- Notez cette <u>nouvelle procédure d'invocation</u> :

```
class Hello {
    public function world($x) { return "hello $x"; }
}

$f = array(new Hello, 'world');

var_dump($f('foo')); // hello foo
```



Short Array Syntax

Python or Javascript anyone ?





Arrays dereferencing

Oui, ça c'était très attendu ...

```
function foo() {
    return array('foo'=>'bar');
}

$var = foo()['foo'];
```



"new" fluent interface

On aime, on n'aime pas, on aime peut-être ...

\$obj = (new MyClass)->someMethod();



Invocations statiques dynamiques

echo Bar::{'foo'}();



WTF?OMG!

- I love that
 - (I really do)

```
class Foo {
   public static function bar() {
       return ['wtf'];
   public function wtf() {
       return 'omg';
$meth = 'bar';
var_dump( (new Foo)->{Foo::$meth()[0]}() ); // omg
```





Notation binaire pour les entiers

- C'est vrai que ça manquait ;-)
 - 0b

```
var_dump(0b1010 + (0b01<<2)); // 14
```

http://julien-pauli.developpez.com/tutoriels/php/bool-op/



Traits

- Mixins, héritage horizontal, héritage multiple
 - Appelez-les comme vous voulez
 - Aimez-les, ou pas ...
- http://www.php.net/traits

```
trait ToArray {
    public function toArray() {
        return $this->array;
    }
}
```



Traits

```
trait ToArray {
    public function toArray() {
        return $this->array;
    }
}
```

```
class MainConfig {
   private $array = ['foo'];
   public function config() { }
}
```

```
class Config extends MainConfig {
   use ToArray;
   private $array = ['bar'];
}
```



```
$config = new Config;
echo $config->toArray()[0]; // bar
```



Closures

- Les closures supportent (enfin) \$this
 - Byebye \$that et autres \$self

```
class A {
  private $value = 1;
  function firstGetter($name) {
     return function() use ($name) {
       return $this->$name;
  function secondGetter($name) {
     return function($name) {
       return $this->$name;
```

```
$a = new A();

$firstGetter = $a->firstGetter('value');

echo $firstGetter(); // 1

$secondGetter = $a->secondGetter();

echo $secondGetter('value'); //1
```



Domaine des closures

 Suite au support de \$this, les closures peuvent maintenant changer de domaine

```
class A {
  private $value = 1;
  function getter() {
     return function() {
       return ++$this->value;};
class B extends A {
  private $value = 42;
```

```
$a = new A;
$b = new B;

$closureA = $a->getter();
echo $closureA(); // 2

$closureB = $closureA->bindTo($b, $b);
echo $closureB(); // 43
```



JSONSerializable

- Nouvelle interface
 - Fonctionnement "à la __toString()"

```
class Foo implements JsonSerializable {
   private $name = 'PHP';

   public function jsonSerialize() {
      return "I love $this->name";
   }
}
```

echo json_encode(new Foo);





PHP a besoin de vous!



Aidez-nous à tester PHP

```
PASS Test scandir() function : error conditions - Incorrect number of args [ext/standard/tests/dir/scandir
SKIP Test scandir() function : error conditions - Non-existent directory [ext/standard/tests/dir/scandir er
PASS Test scandir() function : error conditions - Non-existent directory [ext/standard/tests/dir/scandir er
SKIP Test scandir() function : usage variations - different data types as $dir arg [ext/standard/tests/dir/
PASS Test scandir() function : usage variations - different data types as $dir arg [ext/standard/tests/dir/
PASS Test scandir() function : usage variations - different sorting constants [ext/standard/tests/dir/scand
PASS Test scandir() function : usage variations - diff data types as $sorting order arg [ext/standard/tests
PASS Test scandir() function : usage variations - diff data types as $context arg [ext/standard/tests/dir/s
PASS Test scandir() function : usage variations - different relative paths [ext/standard/tests/dir/scandir
SKIP Test scandir() function : usage variations - different directory permissions [ext/standard/tests/dir/s
SKIP Test scandir() function : usage variations - Wildcards in directory path [ext/standard/tests/dir/scand
PASS Test scandir() function : usage variations - Wildcards in directory path [ext/standard/tests/dir/scand
SKIP Test scandir() function : usage variations - different directory permissions [ext/standard/tests/dir/s
PASS Test scandir() function : usage variations - different file names [ext/standard/tests/dir/scandir vari
PASS Test scandir() function : usage variations - different ints as $sorting order arg [ext/standard/tests/
PASS Directory class behaviour. [ext/standard/tests/directory/DirectoryClass basic 001.phpt]
PASS Directory class behaviour. [ext/standard/tests/directory/DirectoryClass error 001.phpt]
SKIP Test that the Directory extension constants are correctly defined. [ext/standard/tests/directory/direc
PASS Test that the Directory extension constants are correctly defined. [ext/standard/tests/directory/directory
SKIP File type functions [ext/standard/tests/file/001-win32.phpt] reason: only for Windows
PASS File type functions [ext/standard/tests/file/001.phpt]
PASS File/Stream functions [ext/standard/tests/file/002.phpt]
PASS is *() and file exists() return values are boolean. [ext/standard/tests/file/003.phpt]
PASS file put contents() test [ext/standard/tests/file/004.phpt]
PASS Test fileatime(), filemtime(), filectime() & touch() functions : basic functionality [ext/standard/tes
PASS Test fileatime(), filemtime(), filectime() & touch() functions : error conditions [ext/standard/tests/
SKIP Test fileatime(), filemtime(), filectime() & touch() functions : usage variation [ext/standard/tests/f
[EST 6143/8679 [ext/standard/tests/file/005 variation.phpt]
```



Aidez-nous à tester PHP

- Compilez, make test, et envoyez vos résultats
 Do you want to send this report now? [Yns]:
 - http://qa.php.net/reports/?version=5.4.0RC1
 - http://gcov.php.net/viewer.php?version=PHP_5_4
- Lancez vos suites de tests sous PHP 5.4
 - Et faites nous suivre vos remarques
- Participez à l'élaboration de la documentation
 - http://edit.php.net
 - Présentation demain 11h45, salle 2, Yannick Torres



Ressources, infos

- PHP 5.4 : 2011
 - Décembre ? Cadeau de Noël ?
 - On est en RC2 pour le moment
- http://www.php.net/manual/fr/migration54.php
 - En cours d'élaboration
- https://wiki.php.net/todo/php54

- Toutes les RFC en cours :
 - https://wiki.php.net/rfc



Merci!

- Merci à la communauté PHP
- Merci à ses contributeurs
 - Felipe Pena, Etienne Kneuss, Stanislav Malyshev, Gustavo André dos Santos Lopes, David Soria Parra, Christian Stocker, Rob Richards, Pierre Joye, Zeev Suraski, Ilia Alshanetsky [...]
- Merci à vous !

Nous recrutons :)





Questions?

- Chattez avec David Soria Parra
 - De 12h30 à 14h
 - IRC Freenode #phptour
 - http://webchat.freenode.net

