# PHP 5.4とはどんなPHPか？

Yasuo Ohgaki

yohgaki@php.net / yohgaki@ohgaki.net

# 自己紹介

Momonga Linux Project
PostgreSQLユーザ会
日本PHPユーザ会
PHP Group
PHP技術者認定機構 顧問
岡山大学大学院 非常勤講師
エレクトロニック・サービス・イニシアチブ
PROVE for PHP

yohgaki
Yasuo Ohgaki
大垣 靖男

# PHP6はどこに？

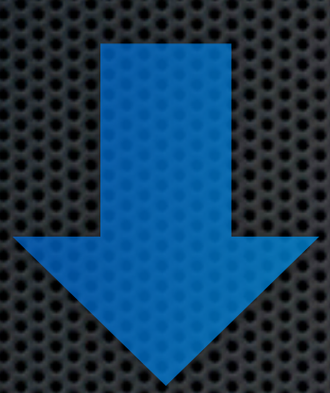
- PHP 6はキャンセル
  - ネイティブUnicodeサポートが目標だった
- PHP 5.4
  - 5.3 + ( PHP6 - Unicodeサポート )

# Unicodeサポートは何処へ？

- ネイティブUnicodeサポートは無い
  （少なくとも当分の間）
- コードが複雑に、速度は低下
  → 開発者の支持が得られなかった

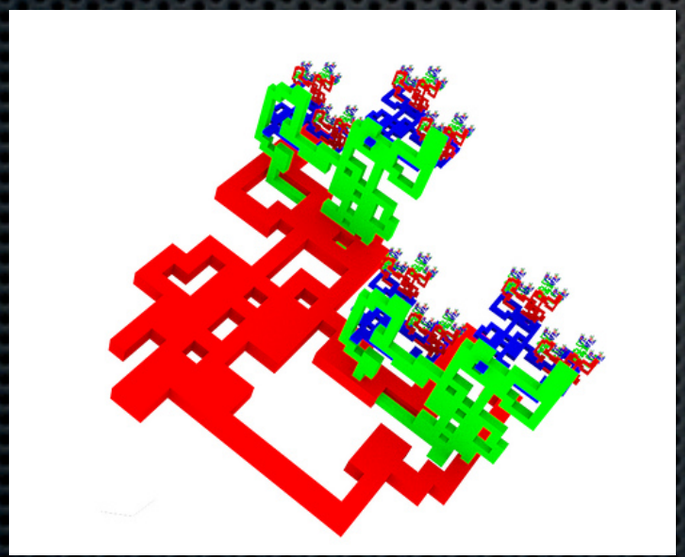
# 仕切り直し

- PHP 5.3ブランチのコードにUnicodeサポートを除いた物をマージ



- PHP 5.4となる
- PHP 5.4はベターPHP 5.3

# 新機能



# 主要な新機能

- Traits
- CLI Server
- “<?”と”<?=”がいつでも利用可能
- 配列文法 [1,2,3]、func()[1]
- (new Foo)->bar(), CLASS::{expr}()
- バイナリ表記　0b00101101101

# Traits

- Class：垂直再利用
  - class foo extends bar {}
- Traits ： 水平利用
  - Classと同様に利用
  - trait foo {}
    class bar { use foo; }
  - Rubyのmixinに相当 （Traitsは静的）

# Traits

```
trait Accessors {
  public function __get($name) {
    if ($this->r_property[$name])
      return $this->$name;
    else
      trigger_error("Access to read protected property");
  }

  public function __set($name, $value) {
    if ($this->w_property[$name])
      $this->$name = $value;
    else
      trigger_error("Access to write protected property");
  }
}
```

https://gist.github.com/1379592

# Traits

```
class OrderLine {
  use Accessors;

  private $r_property = array('price'=>1, 'amount'=>1);
  private $w_property = array('price'=>0, 'amount'=>1);

  protected $price;
  private $amount;
  private $tax = 1.15; // property without getter/setter

  public function getTotal() {
    return $this->price * $this->amount * $this->tax;
  }
}

$line = new OrderLine;

$line->price = 20; //$w_propertyでないのでエラー

$line->amount = 3; //$r_propertyであるのでOK
```

# Traits - 複数使う場合

```
class MyClasss {
  use tool, util {
     tool::escape insteadof util;
     util::escape as escape2;
  }
}
```

# Traits - 可視性

```
class MyClasss {
  use tool, util {
     tool::escape insteadof util;
     util::escape as private escape2;
  }
}
```

# Traits - 再利用

```
trait MyTool {
  use tool, util {
     tool::escape insteadof util;
     util::escape as escape2;
  }
}
```

# Traits - 抽象メソッド

```
trait MyTool {
  use tool, util {
     tool::escape insteadof util;
     util::escape as escape2;
  }
  abstract public function convert();
}
```

# Traits - プロパティ

```
trait MyTrait {
  public $foo = 'abc';
  protected $bar = 'xyz';
}

class MyClass {
  use MyTrait;

  public $foo = 'abc'; // E_STRICT
  protected $bar = '123'; // FATAL ERROR
}
```

# CLI Server

- CLI(コマンドラインPHP)のビルトインHTTPサーバ
  - php -S 127.0.0.1:1080
- コンパクトで速い
- サポートするMIME型はハードコード
- 運用にはWatch Dog必須

# CLI SERVER
# HTMLファイル

Requests per second: 6835.62 [#/sec] (mean)
Time per request: 2.926 [ms] (mean)
Time per request: 0.146 [ms] (mean, across all concurrent requests)
Transfer rate: 827.75 [Kbytes/sec] received

# CLI SERVER
# PHPファイル

Requests per second: **5582.96** [#/sec] (mean)
Time per request: 3.582 [ms] (mean)
Time per request: 0.179 [ms] (mean, across all concurrent requests)
Transfer rate: 665.16 [Kbytes/sec] received

# ショートタグ

- <? ?> <?= ?> が常に利用可能
- <? h(‘<TAG>を含む文字列も安全’) ?>など
- “<?”はXMLのスクリプトタグなので注意が必要

# 配列文法

- $array = [1,2,3];
- $array = [‘a’=>1, ‘b’=>2];
- function foo() {return [‘a’,’b’];};
  echo foo()[1];

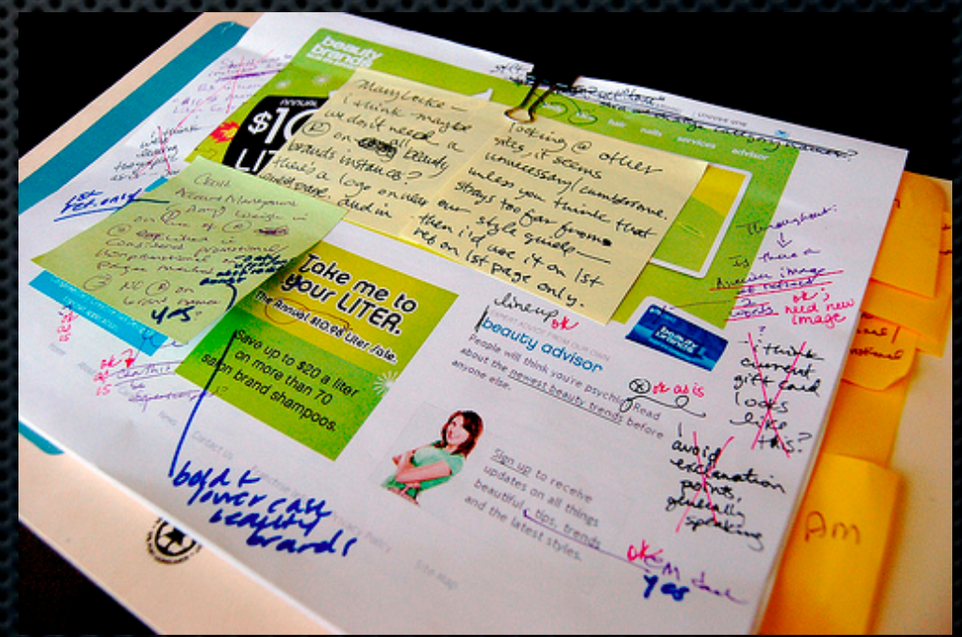
# オブジェクト指向文法

- (new Foo)->bar()
  - 以前からfunc()->method()はサポートされている
- CLASS::{expr}()
  - スタティックな可変関数をサポート
  - $var()は以前からサポート
- Closureで$thisをサポート

# バイナリ表記

- 0b1111 == 0xF == 15
- $int & 0b10101
  などとビットマスクを分り易く表記可能

# 主要な変更箇所


Take me to your LITER.
Save up to $20 a liter on more than 70 salon brand shampoos.
beauty advisor

# ZendMultibyte

- zend.multibyte = on/off
- 従来はコンパイルオプションだった
- SJISなどISO-8859-1非互換マルチバイト文字でソースコードを記述する場合にOn

# マジッククオート

- 入力にaddslashes関数を適用する機能
- そもそも何故こんな有害無益な機能が実装されたのかは不明
- やっと削除される

# セーフモードの削除

- セーフモードはフェイルセーフ機能
  - そもそもサンドボックスをサポートするVMではない
- セキュリティ機能と勘違いする人が多かった
- フェイルセーフ機能としては十分有用だったが削除

# 長い名前の超グローバル

- $HTTP_GET_VARS, $HTTP_POST_VARSなどの長い名前のスーパーグローバル変数が削除
  - メモリ使用量、速度的にはそれほど変化しない
    - PHPの変数はリファレンスカウントをサポート

# breakとcontinue

- break $var;
  - 変数、関数はサポートされない
  - const と同様
  - 実行速度向上に役立つ
  - break 2; などは以前通り

# TZ環境変数

- TZ環境変数を無視
- php.iniのdate.timezone
- date_default_timezone_set()

# 文字列のオフセット

- isset($str[0][0])の戻り値
  - PHP 5.3: false
  - PHP 5.4: false
    - 5.4 RC版ではtrueであったがリリース版は以前と同じ
    - マイグレーションガイドの記述は間違いがあるので注意

# 超グローバルの引数

- スーパーグローバル引数の禁止
- function foo($_GET) などがFatal Error
- そもそも意味がない

# CallbackFilterIterator
# RecursiveCallbackFilterIterator

- Iteratorのイテレーションにコールバック関数を適用し、結果を返す

```
<?php
$dir = new FilesystemIterator(__DIR__);
$files = new CallbackFilterIterator($dir,
    function ($current, $key, $iterator) {
        return $current->isDir() && ! $iterator->isDot();
    }
);
```

# パフォーマンス


U.S. DEPARTMENT OF THE INTERIOR
UTAH
COLORADO
ARIZONA
NEW MEXICO
1992
CADASTRAL SURVEY
BUREAU OF LAND MANAGEMENT

# PHP 5.4のパフォーマンス

- パフォーマンスは全般に向上
- メモリの使用量は低下

| | | | IIS 7.5 | | | | | | Apache 2.2 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | No Cache | | | WinCache | | | No Cache | | | APC -igb |
| | Load Agents | | | | | | | | | | | |
| Application | Physical | Virtual | 5.3.10 | 5.4.0RC7 | gain | 5.3.10 | 5.4.0RC7 | gain | 5.3.10 | 5.4.0RC7 | gain | 5.3 |
| Helloworld | 2 | 8 | 15696.0 | 14977.8 | -4.58 % | 6700.6 | 0 | | 10201.6 | 11316.4 | 10.93 % | 120 |
| | 2 | 16 | 15108.4 | 14649.0 | -3.04 % | 13712.2 | 0 | | 10194.8 | 11189.1 | 9.75 % | 117 |
| | 2 | 32 | 15043.2 | 14510.0 | -3.54 % | 15736.1 | 0 | | 10052.3 | 11288.2 | 12.29 % | 119 |
| Drupal | 2 | 8 | 57.1 | 59.3 | 3.85 % | 203.4 | 0 | | 41.3 | 44.7 | 8.23 % | 121 |
| | 2 | 16 | 56.4 | 58.7 | 4.08 % | 200.7 | 0 | | 40.6 | 44.0 | 8.37 % | 117 |
| | 2 | 32 | 56.7 | 59.0 | 4.06 % | 203.1 | 0 | | 39.2 | 47.4 | 20.92 % | 114 |
| Mediawiki | 2 | 8 | 38.6 | 43.5 | 12.69 % | 70.1 | 0 | | 28.4 | 33.0 | 16.20 % | 52. |
| | 2 | 16 | 38.7 | 44.2 | 14.21 % | 73.4 | 0 | | 28.0 | 32.6 | 16.43 % | 52. |
| | 2 | 32 | 38.9 | 44.4 | 14.14 % | 73.5 | 0 | | 28.1 | 32.4 | 15.30 % | 51. |
| Wordpress | 2 | 8 | 63.0 | 68.2 | 8.25 % | 220.8 | 0 | | 44.7 | 50.0 | 11.86 % | 153 |
| | 2 | 16 | 63.4 | 69.1 | 8.99 % | 219.1 | 0 | | 44.2 | 49.6 | 12.22 % | 152 |
| | 2 | 32 | 62.9 | 68.9 | 9.54 % | 219.8 | 0 | | 43.3 | 49.1 | 13.39 % | 152 |
| Joomla | 2 | 8 | 40.0 | 44.5 | 11.25 % | 60.4 | 0 | | 29.1 | 32.5 | 11.68 % | 36. |
| | 2 | 16 | 39.1 | 43.4 | 11.00 % | 60.9 | 0 | | 28.4 | 32.1 | 13.03 % | 35. |
| | 2 | 32 | 39.6 | 44.2 | 11.62 % | 60.9 | 0 | | 28.9 | 32.2 | 11.42 % | 35. |
| Symfony | 2 | 8 | 29.4 | 29.8 | 1.36 % | 34.3 | 0 | | 10.1 | 10.3 | 1.98 % | 0 |
| | 2 | 16 | 26.8 | 27.3 | 1.87 % | 30.6 | 0 | | 9.8 | 10.0 | 2.04 % | 0 |
| | 2 | 32 | 28.8 | 30.1 | 4.51 % | 36.7 | 0 | | 10.7 | 11.7 | 9.35 % | 0 |

- http://windows.php.net/downloads/snaps/ostc/pftt/perf/results-20120203-5.3.10-5.4.0RC7.html

# abによるベンチ


# Linux/Apache 2.4.1 HTML

Requests per second: 12870.22 [#/sec] (mean)
Time per request: 1.554 [ms] (mean)
Time per request: 0.078 [ms] (mean, across all concurrent requests)
Transfer rate: 3242.69 [Kbytes/sec] received

# abによるベンチ

# PHP 5.3.11-dev

Requests per second: 6354.39 [#/sec] (mean)
Time per request: 3.147 [ms] (mean)
Time per request: 0.157 [ms] (mean, across all concurrent requests)
Transfer rate: 1197.65 [Kbytes/sec] received

# abによるベンチ

# PHP 5.4.1RC1-dev

Requests per second:　9954.96 [#/sec] (mean)
Time per request:　2.009 [ms] (mean)
Time per request:　0.100 [ms] (mean, across all concurrent requests)
Transfer rate:　2061.05 [Kbytes/sec] received

http://svn.php.net/viewvc/php/php-src/branches/PHP_5_4/Zend/micro_bench.php?view=co

```
C:\php538>php bench.php
empty_loop          0.347
func()              1.293    0.946
undef_func()        1.358    1.011
int_func()          1.223    0.876
$x = self::$x       0.924    0.577
self::$x = 0        0.895    0.548
isset(self::$x)     0.874    0.527
empty(self::$x)     0.889    0.542
$x = Foo::$x        2.050    1.703
Foo::$x = 0         2.235    1.888
isset(Foo::$x)      2.744    2.397
empty(Foo::$x)      2.051    1.704
self::f()           1.958    1.611
Foo::f()            3.501    3.154
$x = $this->x       0.918    0.571
$this->x = 0        1.127    0.780
$this->x += 2       0.895    0.548
++$this->x          0.778    0.431
--$this->x          0.778    0.431
$this->x++          0.959    0.612
$this->x--          0.977    0.630
isset($this->x)     0.899    0.552
empty($this->x)     0.929    0.582
$this->f()          2.143    1.796
$x = Foo::TEST      0.789    0.442
new Foo()           5.202    4.855
$x = TEST           0.747    0.400
$x = $_GET          0.844    0.497
$x = $GLOBALS['v'] 0.988     0.641
$x = $hash['v']     0.695    0.348
$x = $str[0]        1.796    1.449
$x = $a ?: null     6.258    5.911
$x = $f ?: tmp      0.821    0.474
$x = $f ? $f : $a   6.045    5.698
$x = $f ? $f : tmp 0.830     0.483
---------------------
Total          57.764
```

http://pastebin.com/raw.php?i=52s1kzMa

```
C:\php54>php bench.php
empty_loop          0.264
func()              1.147    0.883
undef_func()        1.152    0.887
int_func()          0.941    0.677
$x = self::$x       0.635    0.371
self::$x = 0        0.629    0.365
isset(self::$x)     0.581    0.317
empty(self::$x)     0.560    0.296
$x = Foo::$x        0.561    0.296
Foo::$x = 0         0.527    0.263
isset(Foo::$x)      0.486    0.222
empty(Foo::$x)      0.485    0.221
self::f()           1.363    1.099
Foo::f()            1.147    0.883
$x = $this->x       0.529    0.265
$this->x = 0        0.797    0.533
$this->x += 2       0.566    0.301
++$this->x          0.468    0.204
--$this->x          0.465    0.201
$this->x++          0.512    0.248
$this->x--          0.512    0.248
isset($this->x)     0.528    0.264
empty($this->x)     0.560    0.296
$this->f()          1.402    1.138
$x = Foo::TEST      0.583    0.319
new Foo()           2.557    2.293
$x = TEST           0.495    0.231
$x = $_GET          0.667    0.402
$x = $GLOBALS['v'] 0.840     0.576
$x = $hash['v']     0.565    0.301
$x = $str[0]        1.007    0.743
$x = $a ?: null     5.910    5.646
$x = $f ?: tmp      0.536    0.272
$x = $f ? $f : $a   6.005    5.740
$x = $f ? $f : tmp 0.566     0.301
---------------------
Total          36.548
```

http://pastebin.com/raw.php?i=1AU10Adf

# PHP 5.4のZend/bench.php

```
[yohgaki@dev php-src]$ cat bench-5.3.11dev.txt
simple               0.250
simplecall           0.407
simpleucall          0.390
simpleudcall         0.390
mandel               0.769
mandel2              0.766
ackermann(7)         0.414
ary(50000)           0.047
ary2(50000)          0.043
ary3(2000)           0.349
fibo(30)             1.187
hash1(50000)         0.077
hash2(500)           0.067
heapsort(20000)      0.205
matrix(20)           0.208
nestedloop(12)       0.432
sieve(30)            0.198
strcat(200000)       0.025
----------------------
Total                6.226
```

```
[yohgaki@dev php-src]$ cat bench-5.4.1rc1.txt
simple               0.192
simplecall           0.346
simpleucall          0.360
simpleudcall         0.368
mandel               0.479
mandel2              0.584
ackermann(7)         0.322
ary(50000)           0.046
ary2(50000)          0.045
ary3(2000)           0.311
fibo(30)             0.984
hash1(50000)         0.070
hash2(500)           0.065
heapsort(20000)      0.168
matrix(20)           0.185
nestedloop(12)       0.317
sieve(30)            0.179
strcat(200000)       0.023
----------------------
Total                5.043
```

ただし、必ずしも速くなる

とは限らない

# abによるベンチ

# PHP 5.3.11-dev Dokuwiki

Requests per second:  **51.07** [#/sec] (mean)
Time per request:  391.582 [ms] (mean)
Time per request:  19.579 [ms] (mean, across all concurrent requests)
Transfer rate:  526.45 [Kbytes/sec] received

# abによるベンチ

# PHP 5.4.1-dev Dokuwiki

Requests per second: 47.15 [#/sec] (mean)
Time per request: 424.209 [ms] (mean)
Time per request: 21.210 [ms] (mean, across all concurrent requests)
Transfer rate: 485.87 [Kbytes/sec] received

# その他

# Windowsユーザ

- PHP 5.4はWindows XPとWindows Server 2003をサポートする最後のPHP

# PHP 5.3はどうなる？

- 恐らく1年後にサポート終了（EOL）のアナウンスがある。
- 事前アナウンスは多分ない。
- EOL1年間はセキュリティパッチが提供される。

# 次のPHPは

- 次のPHPは恐らくPHP 5.5
- リリース時期は恐らく2014春以降

# PHP 5.4のライフタイム

- PHP 5.5リリースの1年後にEOL、その更に1年後にセキュリティパッチ提供も終了
- PHP 5.5は2年後にリリース予定
- PHP 5.4がサポートされる期間は最小4年
  - 2016年春まで。長くて17年春頃まで
  - 新バージョンリリースが延びる事はよく有る
- 移行する人は直ぐに移行する方が良い


# ご清聴ありがとう
# ございました。