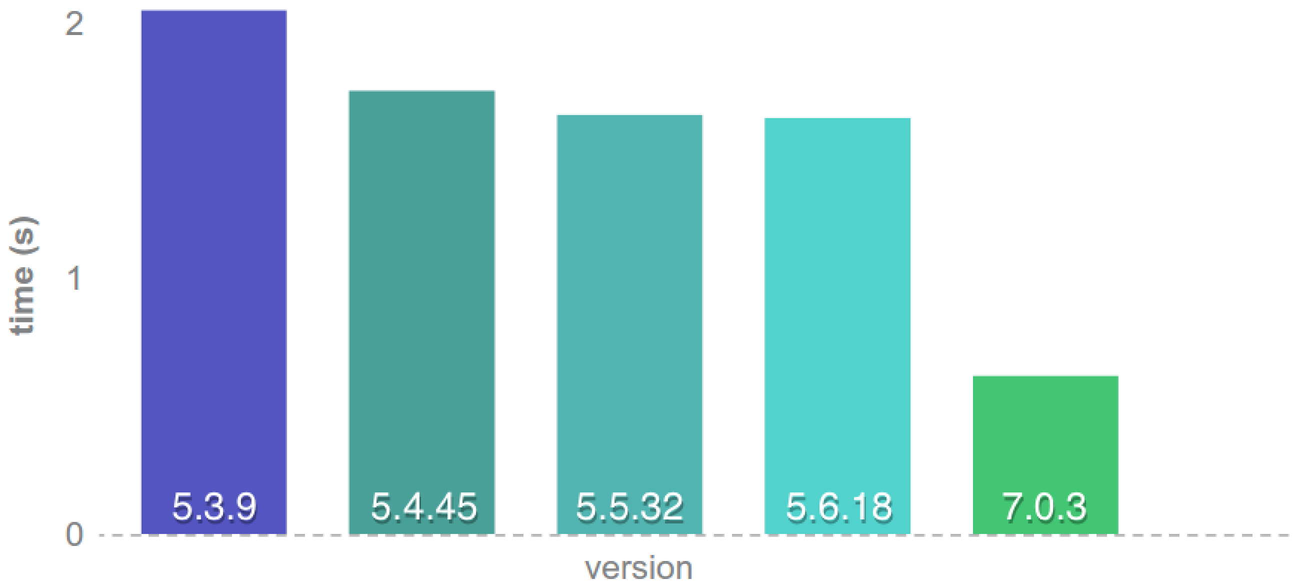
The low down on PHP 7

Rob Allen

19ft.com ~ @akrabat ~ February 2016

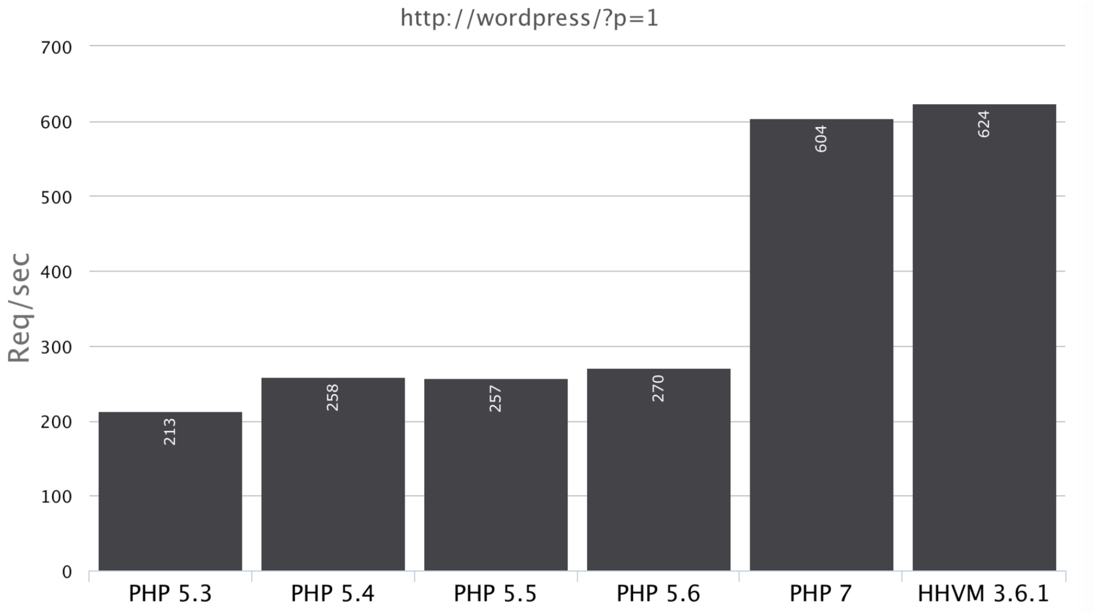
PHP 7 is *fast* !

bench.php



Source: Lorna Mitchell ~ http://www.lornajane.net

Wordpress 4.1.1



Source: Rasmus Lerdorf ~ http://talks.php.net/fluent15#/

How did we get it so fast?

- Zend's Dmitry Stogov spearheaded phpng project
- Significant reworking of the Zend Engine
    - Compacted data structures
    - New memory manager
    - Faster function parameter parsing
    - Many other small improvements!
- All extensions need updating

Abstract syntax trees

- Better internal structure
- Performance opportunities
- Static analysis tools possible

New features

Type hints

PHP 5

class name

interface name

self

`array`

`callable`

```
function shorten(array $data, $length) {
   return array_slice($data, -$length);
}
```

Type hints

PHP 5

class name

interface name

self

array

callable

PHP 7

bool

float

int

string

```
function shorten(array $data, int $length) {
   return array_slice($data, -$length);
}
```

Call the method

```
$array = [10, 11, 12, 13, 14];
$new = shorten($array, "2");

print_r($new);
// Array
// (
//     [0] => 10
//     [1] => 11
//     [2] => 12
// )
```

Strict type checks

Enable in the **calling context**:

```
<?php
declare(strict_types=1); // enable strict type check

$array = [10, 11, 12, 13, 14];
$new = shorten($array, "2");
print_r($new);
```

```
PHP Fatal error:  Uncaught TypeError: Argument 2 passed to shorten() must
be of the type integer, string given
```

Return type hints

Declare the return type after the parameter list

```
function shorten(array $data, int $length) : array
{
    if ($length > 0) {
        return array_slice($data, 0, -$length);
    }
    return null;
}

$array = [10, 11, 12, 13, 14];
$new = shorten($array, 2);

// [10, 11, 12]
```

Return type hints

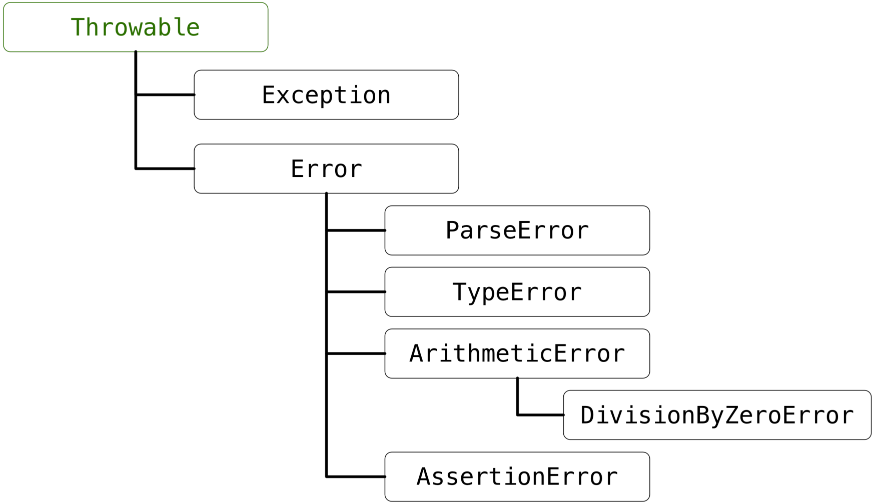
Cannot return any other type, including `null`

```
$array = [10, 11, 12, 13, 14];
$new = shorten($array, -1);
```

```
PHP Fatal error:  Uncaught TypeError: Return value of shorten() must be
of the type array, null returned
```

Exceptions and Errors

Some PHP internal errors are catchable via the `Error` class



Catching Exceptions and errors

```
try {
    $array = [10, 11, 12, 13, 14];
    $new = shorten($array, "2");
} catch (Exception $e) {
    echo "An exception occurred";
} catch (TypeError $e) {
    echo "you passed the wrong type of argument";
} catch (Error $e) {
    echo "A PHP error occurred";
}
```

Anonymous classes

```
$bigInt = new class extends Zend\Math\BigInteger {
    public function base10ToBase2($number) {
        return $this->baseConvert($number, 10, 2);
    }
};

$binary = $bigInt->base10ToBase2($aLongNumber);
```

Null Coalesce Operator

Operator `??` is ternary shorthand (`?:`) but with `isset()`.

```
$q = $_GET['q'] ?? '';

$year = $_POST['year'] ?? $_GET['year'] ?? 2016;
```

Spaceship operator

AKA: Combined comparison operator

```
echo 1 <=> 1; // 0
echo 1 <=> 2; // -1
echo 2 <=> 1; // 1
```

Works for integers, floats, strings, arrays & objects

```
// practical use case: sort by multiple properties
$array = [$user1, $user2, $user3, $user4];
usort($array, function ($l, $r) {
    return [$l->surname, $l->forename] <=> [$r->surname, $r->forename];
});
```

Upgrading challenges

Uniform variable syntax

More consistent and complete variable syntax with fast parsing

but subtle changes from PHP 5 when dereferencing or using `$$`
Solution is to add more `{` and `}`

e.g.

```
// PHP 5                     // PHP 7
global $$foo->bar;           global ${$foo->bar};
```

Static analyser: https://github.com/etsy/phan

Foreach changes

- The internal array pointer no longer moves
  - `current()` & `next()` no longer work within foreach loop
  - `current($array)` is zeroth element after iteration
- Array changes within `foreach()` by value are now ignored for iteration.
  - can no longer `unset()` future elements as you go to skip them

RFC: https://wiki.php.net/rfc/php7_foreach

Hex numbers in strings

Hex numbers in strings are no longer detected when converting to numbers

```
$num = 1 + "0xFF";

// in PHP 5: 256
// in PHP 7: 1
```

Deprecated Features

If it was `E_DEPRECATED` in PHP 5 then it'll probably be removed soon!

Note:

- The RFC to remove things was agreed but it hasn't been implemented yet
- The `mysql_*` functions really are removed
- PHP 4 constructors are special!

The Future

- PHP 5.6 security support has been extended
    - Support until 31st December 2016
    - Security fixes until 31st December 2018

- PHP 7.0 is safe to run
- PHP 7.1 looks even better
    - void return type
    - HTTP/2 server push
    - Class constant visibility modifiers

Upgrade… it's *very fast*!

Thank you!

Rob Allen ~ 19ft.com ~ @akrabat