

PHP7: The Big Five

cal@zend.com

zend.com

zend[®]

Requests for Comment

- A brief history of time
- What are RFCs
- How I have classified the RFCs
 - High Impact
 - BC Breaks

The Big 5

1. Exceptions in the engine
2. Scalar Type Declarations
3. Combined Comparison (Spaceship) Operator
4. Easy User-land CSPRNG
5. Null Coalesce Operator

PHP 7: The Big Five

1: Exceptions in the
Engine

Exceptions in the engine/ Throwable Interface

Problem

- Fatal errors Cannot be gracefully handled
- Fatal errors do not invoke a finally block
- Fatal errors do not call an object's destructor

```
function do_something($obj) {  
    $obj->nope();  
}
```

```
do_something(null); // oops!
```

Exceptions in the engine/ Throwable Interface

Solution

```
try  {
    do_something(null);
} catch (\Error $e) {
    echo "Error: {$e->getMessage()}\n";
}

// Error: Call to a member function
method() on a non-object
```

Exceptions in the engine/ Throwable Interface

The new interface **Throwable**

- Interface **Throwable**
 - **Exception** implements **Throwable**
 - **Error** implements **Throwable**
 - **TypeError** extends **Error**
 - **ParseError** extends **Error**

Exceptions in the engine/ Throwable Interface

Do not catch Errors except for logging and cleanup.
Errors are code issues that should be fixed, not
conditions that can be handled at runtime.

Exceptions in the engine/ Throwable Interface

- getMessage()
- getCode()
- getFile()
- getLine()
- getTrace()
- getTraceAsString()
- __toString()

Exceptions in the engine/ Throwable Interface

```
function add(int $left, int $right) {  
    return $left + $right;  
}  
  
try {  
    echo add('left', 'right');  
} catch (\TypeError $e) {  
    // Log error and end gracefully  
} catch (\Exception $e) {  
    // Handle any exceptions  
} catch (\Throwable $e) {  
    // Handle anything else  
}
```

Exceptions in the engine/ Throwable Interface

ParseError

- Thrown when you have a parse error in your code
- Will not allow bad code to run at compile time.
- Will be thrown if you have a parse error in an eval()
- Will be thrown if you have an error in a file that is included during execution.

Exceptions in the engine/ Throwable Interface

ParseError

```
$code = 'var_dup($admin);'

try {
    $result = eval($code);
} catch (\ParseError $error) {
    // Handle $error
}
```

Exceptions in the engine/ Throwable Interface

ParseError

```
try {
    if ($admin) {
        include "./this_code_has_issues.php";
    }
} catch (\ParseError $error) {
    // Handle $error
}
```

Exceptions in the engine/ Throwable Interface

Missing Class

```
<?php
require __DIR__ . '/vendor/autoload.php';

try {
    $x = new ClassDoesNotExist();
} catch (\Error $e) {
    print_r($e);
} finally {
    Echo "\nDone\n";
}
```

Exceptions in the engine/ Throwable Interface

PHP 5.6

```
calevans: ~/work $ php test.php
```

```
PHP Fatal error:  Class 'MyClass' not found  
in /home/calevans/work/test.php on line 5
```

Exceptions in the engine/ Throwable Interface

PHP7

```
calevans: ~/work $ php ./test.php  
Error Object
```

```
(  
  
[message:protected] => Class 'MyClass' not found  
[string:Error:private] =>  
[code:protected] => 0  
[file:protected] => /home/calevans/work/test.php  
[line:protected] => 5  
[trace:Error:private] => Array  
 (   
 )  
  
[previous:Error:private] =>  
)
```

Done

```
calevans: ~/work $
```

Exceptions in the engine/ Throwable Interface

Backwards Compatibility Break

- Old: Parse errors generated during eval() (but not require etc) are non-fatal.
- New: eval() now throws an Error

Exceptions in the engine/ Throwable Interface

Backwards Compatibility Break

- E_RECOVERABLE_ERROR
Can't ignore these anymore with the error handler.

Exceptions in the engine/ Throwable Interface

Backwards Compatibility Break

- **Throwable**
- **Error**
- **TypeError**
- **ParseError**

Exceptions in the engine/ Throwable Interface

Benefits

- `finally` gets called
- `__destroy()` gets called.
- Fully backwardly compatible

Exceptions in the engine/ Throwable Interface

Issues

- Error & ParseError implement Throwable and are the new Catchable exceptions

Exceptions in the engine/ Throwable Interface

Issues

- E_ERROR
- E_RECOVERABLE_ERROR
- E_PARSE
- E_COMPILE_ERROR

All now converted to **Error**

Exceptions in the engine/ Throwable Interface

Issues

- Discourages introducing new errors of the type E_ERROR or E_RECOVERABLE_ERROR.

PHP 7: The Big Five

2: Scalar Type Declarations

Scalar Type Declarations

- int
- float
- string
- bool

Scalar Type Declarations

```
function add($a, $b) {  
    return $a + $b;  
}  
  
var_dump(add(1, 2)); // int(3)  
  
var_dump(add(1.5, 2.5)); // float(4)  
  
var_dump(add("1 foo", "2")); // int(3)
```

Scalar Type Declarations

```
declare(strict_types=1);
```

Scalar Type Declarations

```
declare(strict_types=1);

function add(float $a, float $b) {
    return $a + $b;
}

var_dump(add(1.5, 2.5)); // float(4)

var_dump(add("1 foo", "2")); // TypeError

var_dump(add(1, 2)); // float(3) . . . but why?
```

Scalar Type Declarations

Widening

- The only type casting done in strict mode
- Integers can be “widened” into floats.

```
declare(strict_types=1);

function add(float $a, float $b): float {
    return $a + $b;
}

var_dump(add(1, 2)); // float(3)
```

Scalar Type Declarations

```
declare(strict_types=1);
```

Scalar Type Declarations

Return Type Declarations

```
function foobar(): int {  
    return 1;  
}
```

Scalar Type Declarations

Not Allowed

- You cannot change the return type of a subclassed method.

```
Class MyClass
{
    public function foo(): array {
        return [];
    }
}
```

```
Class MyOtherClass extends MyClass
{
    public function foo(): MyClass {
        return new MyClass();
    }
}
```

Scalar Type Declarations

ALLOWED

```
Class MyClass {  
    function make(): MyClass  
    {  
        return new MyClass();  
    }  
  
Class MyOtherClass extends MyClass {  
  
    function make(): MyOtherClass  
    {  
        return new MyOtherClass();  
    }  
}
```

PHP 7: The Big Five

3: Combined
Comparison
(Spaceship) Operator

<=>

Combined Comparison (Spaceship) Operator

- PHP's first trinary operator

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

Combined Comparison (Spaceship) Operator

Operator	<=> Equilivant
<code>\$a < \$b</code>	<code>(\$a <=> \$b) === -1</code>
<code>\$a <= \$b</code>	<code>(\$a <=> \$b) === -1 (\$a <=> \$b) === 0</code>
<code>\$a == \$b</code>	<code>(\$a <=> \$b) === 0</code>
<code>\$a != \$b</code>	<code>(\$a <=> \$b) !== 0</code>
<code>\$a >= \$b</code>	<code>(\$a <=> \$b) === 1 (\$a <=> \$b) === 0</code>
<code>\$a > \$b</code>	<code>(\$a <=> \$b) === 1</code>

Combined Comparison (Spaceship) Operator

Examples

Strings

```
echo "a" <=> "a"; // 0
echo "a" <=> "b"; // -1
echo "b" <=> "a"; // 1
echo "a" <=> "aa"; // -1
echo "zz" <=> "aa"; // 1
```

Combined Comparison (Spaceship) Operator

Examples

Arrays

```
echo [] <=> [] ; // 0
```

```
echo [1, 2, 3] <=> [1, 2, 3] ; // 0
```

```
echo [1, 2, 3] <=> [] ; // 1
```

```
echo [1, 2, 3] <=> [1, 2, 1] ; // 1
```

```
echo [1, 2, 3] <=> [1, 2, 4] ; // -1
```

Combined Comparison (Spaceship) Operator

Examples

Objects

```
$a = (object) ["a" => "b"];  
$b = (object) ["a" => "b"];  
echo $a <=> $b; // 0
```

```
$a = (object) ["a" => "b"];  
$b = (object) ["a" => "c"];  
echo $a <=> $b; // -1
```

```
$a = (object) ["a" => "c"];  
$b = (object) ["a" => "b"];  
echo $a <=> $b; // 1
```

Combined Comparison (Spaceship) Operator

usort() example

```
$array = ['oranges',
          'apples',
          'bananas',
          'grapes'];

usort($array,
      function ($left, $right)
{
    return $left<=>$right;
} );
print_r($array);
Array
(
    [0] => apples
    [1] => bananas
    [2] => grapes
    [3] => oranges
)
```

PHP 7: The Big Five

4: Easy User-land
CSPRNG

The Big 5

Easy User-land CSPRNG

- Reliable, user land **C**ryptographically **S**ecure
Pseudo**R**andom **N**umber **G**enerator
- No easy way to access cryptographically strong random numbers in user-land.
 - CryptGenRandom on Windows
 - /dev/urandom on Linux/OSX
- Users may attempt to generate their own streams of random bytes...and this is something we absolutely want to avoid.

Easy User-land CSPRNG

```
$random = fread(fopen('/dev/urandom', 'r'),  
16);
```

Easy User-land CSPRNG

Two new functions

```
random_bytes(int length);  
$randomStr = random_bytes(16);
```

```
random_int(int min, int max);  
$randomInt = random_int(1, 20);
```

Easy User-land CSPRNG

Possible BC Break!

- New Reserved Method Names
 - random_bytes
 - random_int

PHP 7: The Big Five

5: Null Coalesce
Operator

The Big 5

Null Coalesce Operator

- Allow developers to test a variable and return either it's value or a default value.
- A common construct for PHP developers is this command:

```
$fullName .= isset($firstName) ? $firstName : "Cal";
```

This will do the job, but is cumbersome.

Null Coalesce Operator

```
$name = $firstName ?? "Cal";  
$name .= " ";  
$name .= $lastName ?? "Evans";
```

`$name` will always contain a string value, even if `$firstName` or `$lastName` are null

The Big 5

Null Coalesce Operator

```
$myName      = 'Cal';
$username   = $myName ?? 'nobody'; // Cal

unset($myName);

$username = $myName ?? 'nobody'; // nobody
```

Null Coalesce Operator

```
$x = NULL;  
$y = NULL;  
$z = 3;  
var_dump($x ?? $y ?? $z); // int(3)
```

RFCs

1. https://wiki.php.net/rfc/engine_exceptions_for_php7
2. https://wiki.php.net/rfc/scalar_type_hints_v5
3. https://wiki.php.net/rfc/easy_userland_csprng
4. <https://wiki.php.net/rfc/csrandombytes>
5. https://wiki.php.net/rfc/isset_ternary

Wrap Up

...one more thing

PHP 7: The Big Five

Move the phpng branch into master

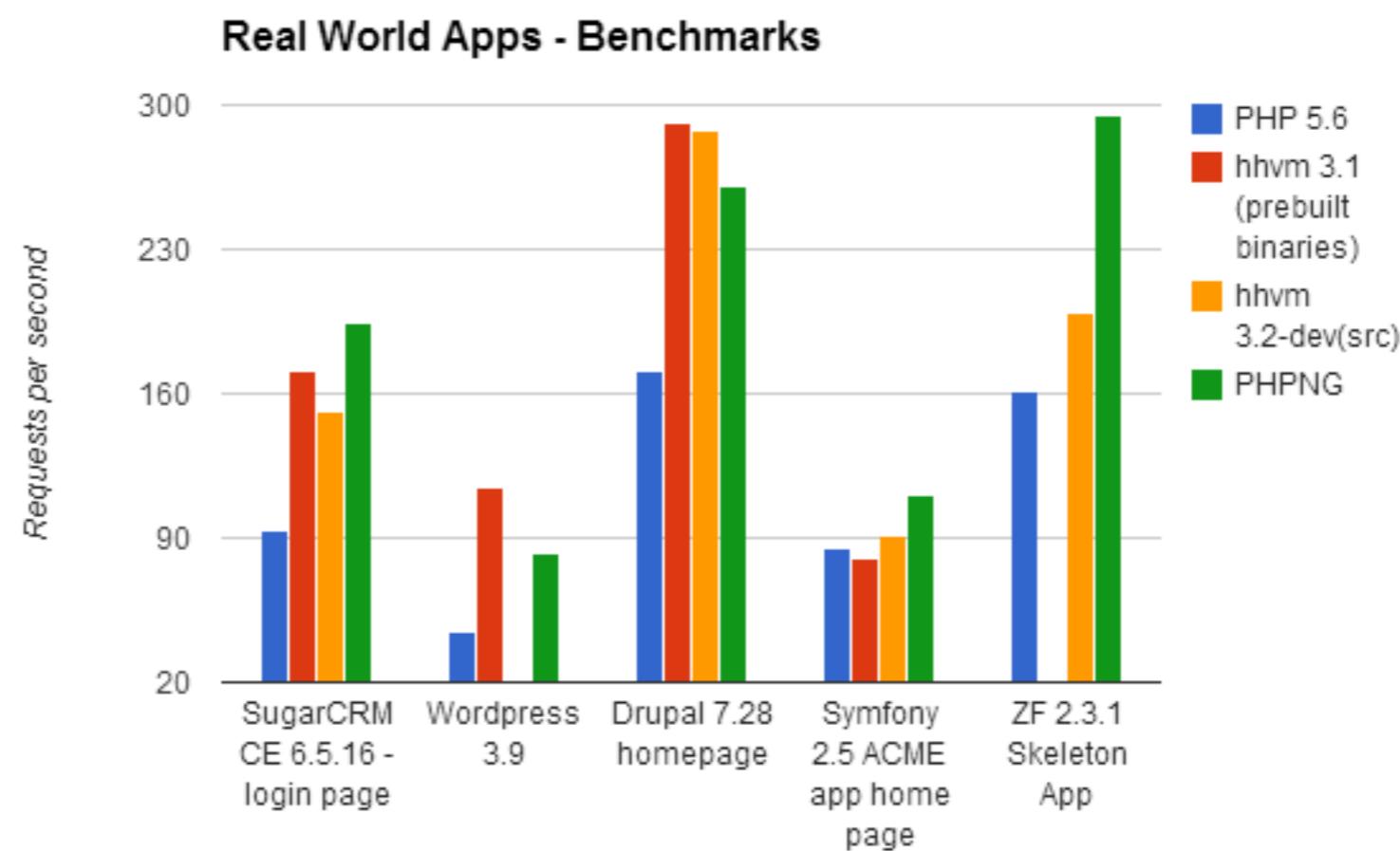
<https://wiki.php.net/rfc/phpng>

Authors : Dmitry Stogov, Zeev Suraski, and team

Move the phpng branch into master

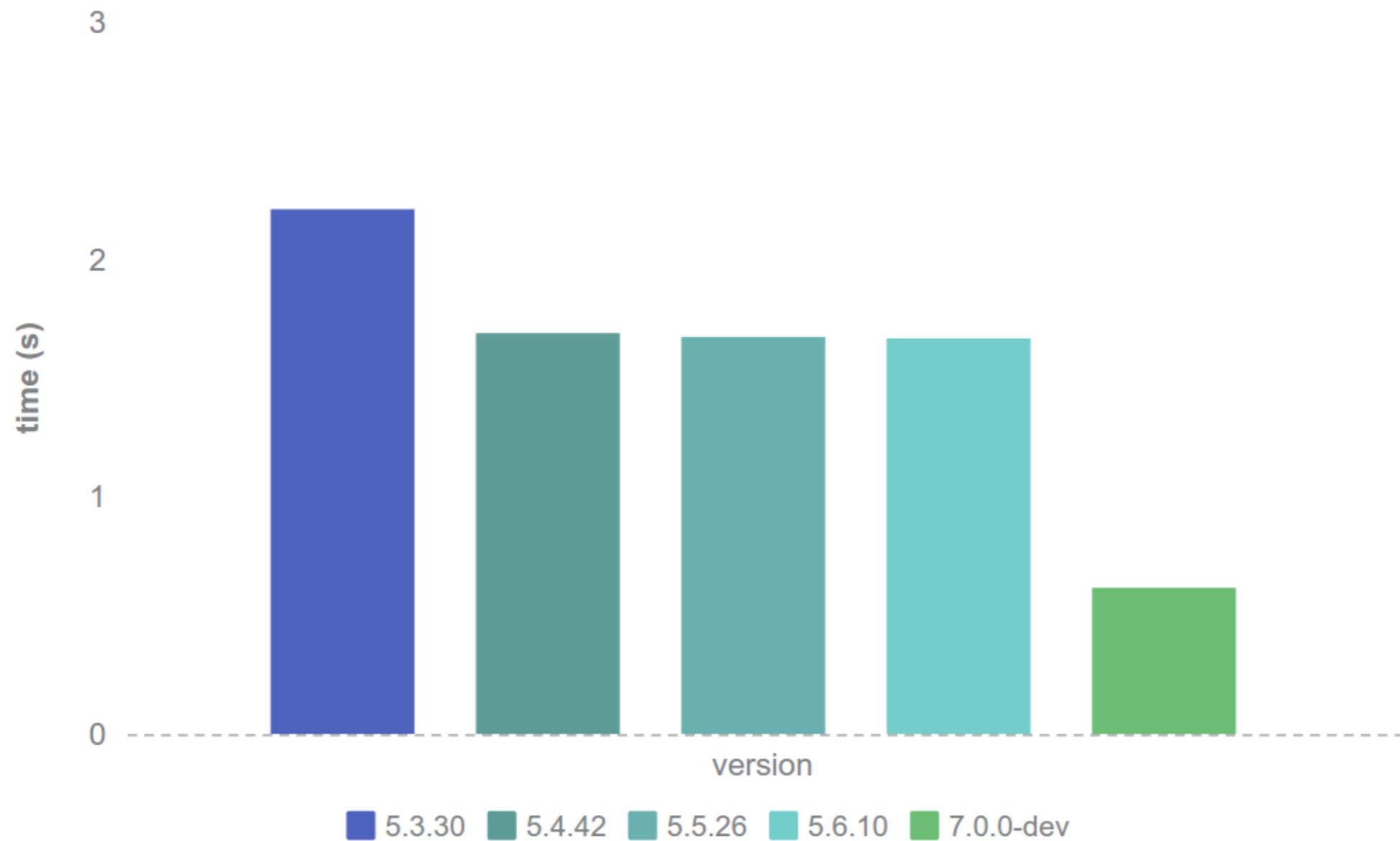
- ~95% faster than 5.6
[http://www.reddit.com/r/PHP/comments/305ck6/
real_world_php_70_benchmarks/](http://www.reddit.com/r/PHP/comments/305ck6/real_world_php_70_benchmarks/)
- ~85% faster than 5.6
<http://zsuraski.blogspot.com/2014/07/benchmarking-phpng.html>
- Drupal is 58% faster on PHP7 vs. PHP 5.6
<http://www.drupalonwindows.com/en/blog/benchmarking-drupal-7-php-7-dev>

Move the phpng branch into master



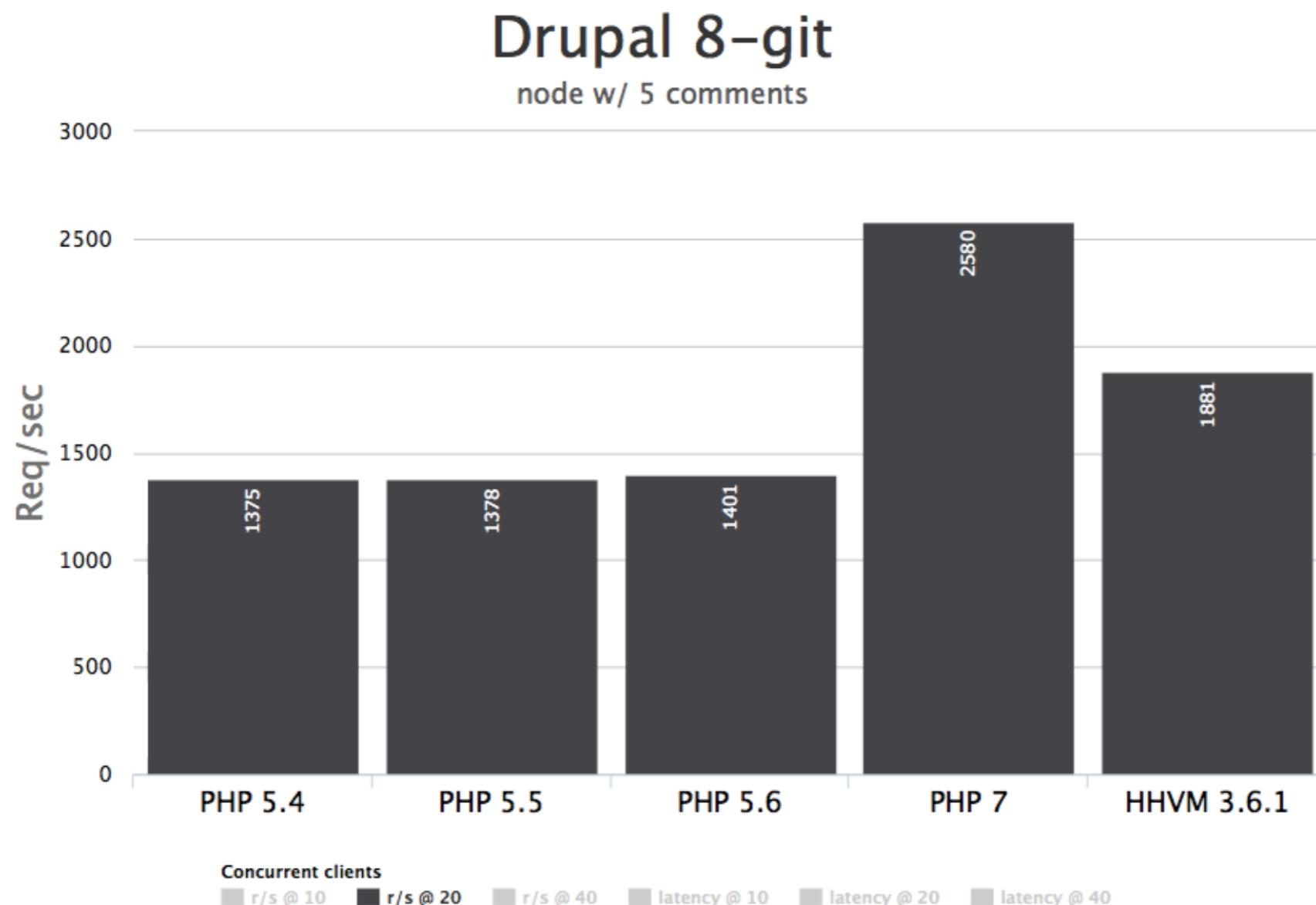
<http://zsuraski.blogspot.com/2014/07/benchmarking-phpng.html>

Move the phpng branch into master



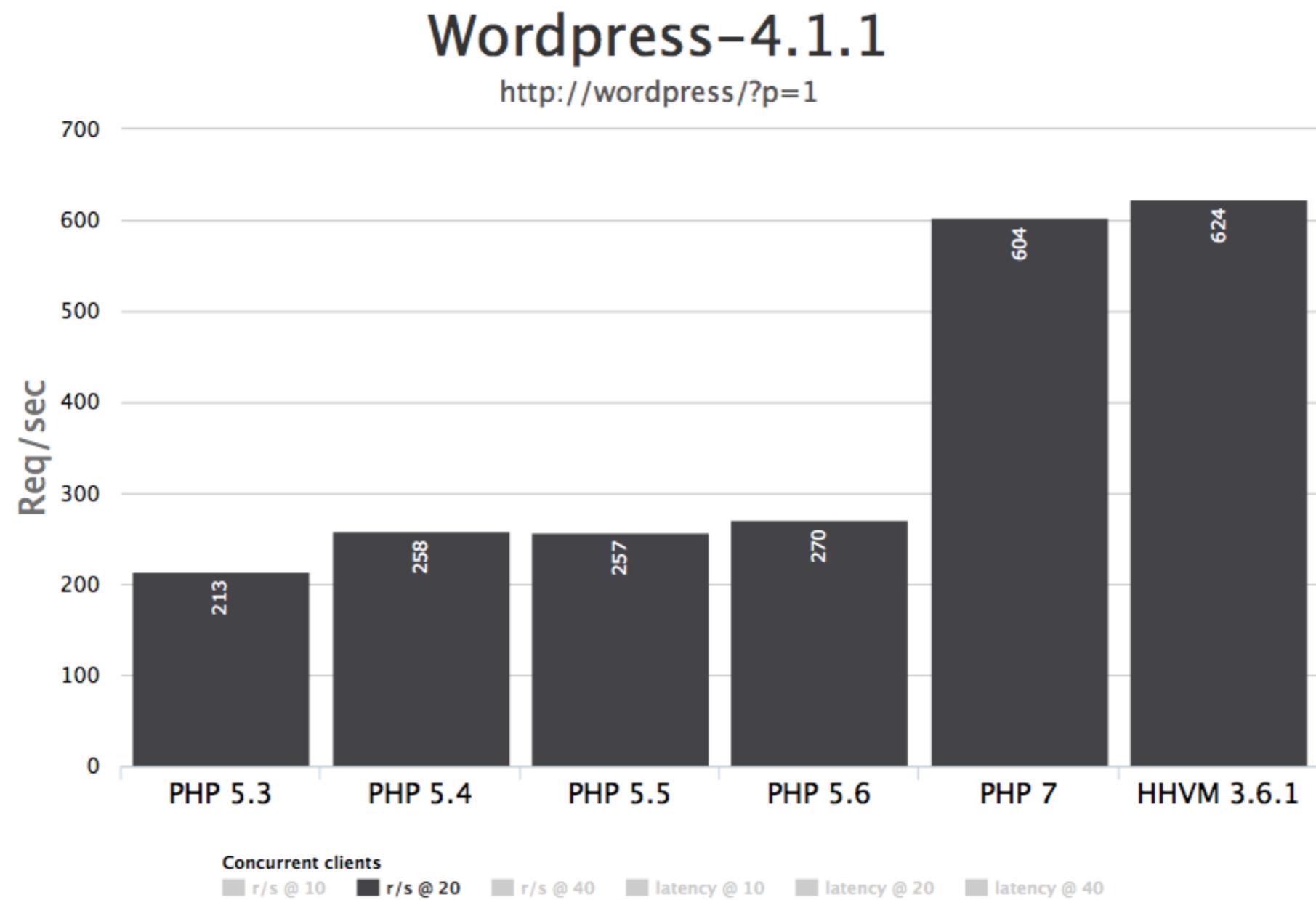
- <http://www.lornajane.net/posts/2015/php-7-benchmarks>

Move the phpng branch into master



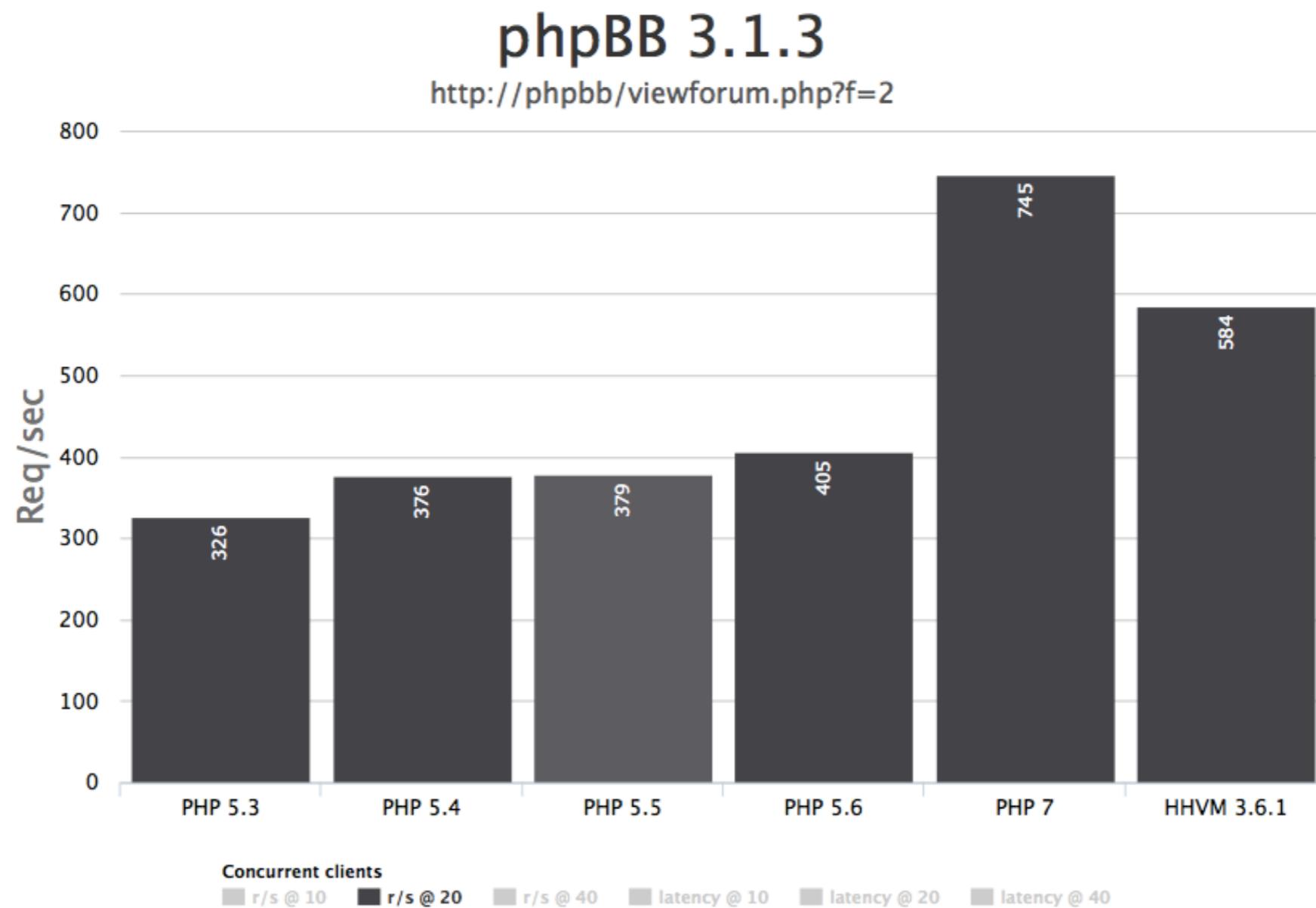
<http://talks.php.net/fluent15#/>

Move the phpng branch into master



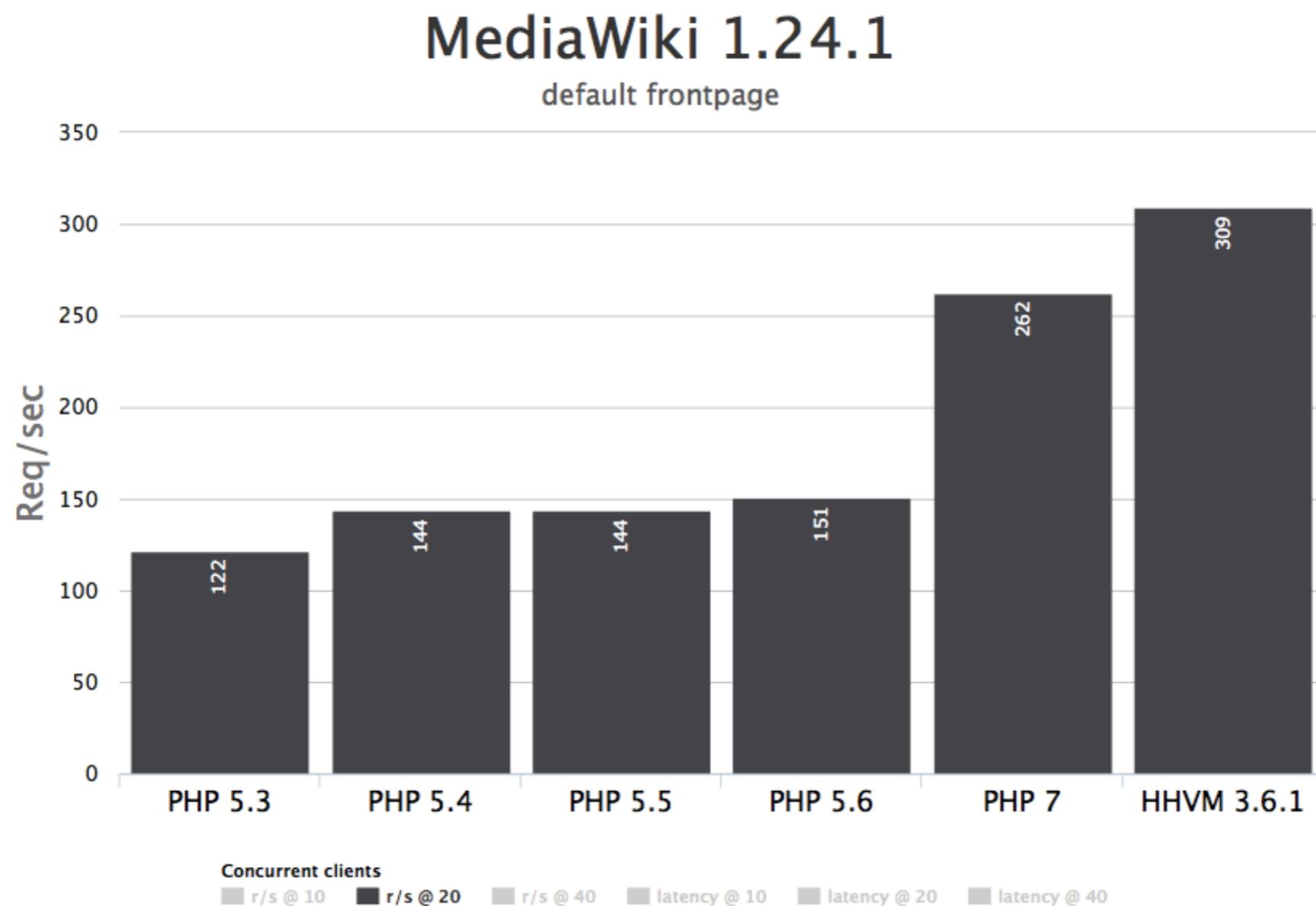
<http://talks.php.net/fluent15#/>

Move the phpng branch into master



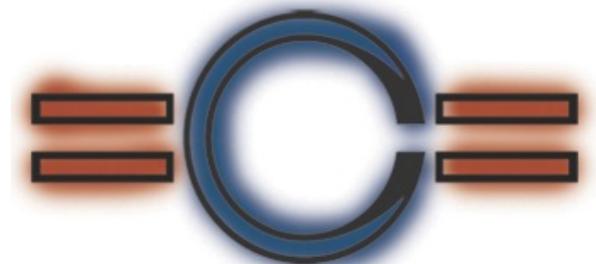
<http://talks.php.net/fluent15#/>

Move the phpng branch into master



<http://talks.php.net/fluent15#/>

Thank you



Cal Evans



- b. blog.calevans.com
- t. [@calevans](https://twitter.com/calevans)
- w. devzone zend.com
- e. cal.e@zend.com



Want to know more?
zend.com/php7-training

