

Forum:[Hack Forums](#)

Thread: [What Features Would You Like to See in PHP?](#)

Haxalot - September 24, 2017, 11:25 am

I'm looking to hear what features/changes developers here would like to see in/to the PHP language.

For example, features such as: Typed properties

Short closure syntax

Async primitives

Enums

Generics

Changes that I don't want to hear about (because it's highly unlikely they'll ever happen):
Standardising function names and parameter orders

Language-level support for multithreading

So, excluding the lists above, are there any features from other languages that you would like to see in PHP? Or perhaps any changes you would like to see to the language (be it syntax or semantics)?

Sakuto - September 24, 2017, 12:35 pm

Asynchronous request.

And everything else you stated. They are basically the reason of why I've migrated to C#.

Haxalot - September 24, 2017, 1:51 pm

(Sep 24, 2017 12:35 PM)Sakuto Wrote: [\[CITE\]](#) Asynchronous request.

And everything else you stated. They are basically the reason of why I've migrated to C#.

I believe async IO will come eventually, but there are several difficulties that need to be sorted out first. I don't see it coming until after PHP 7.3, at the very earliest.

I believe the other items I've listed are currently being worked on, which is why I asked here for other ideas, since I'm looking for things to work on myself. I think pattern matching in switch statements may be a worthwhile feature to have, so maybe I'll look into that...

(Sep 24, 2017 12:59 PM)Yani Wrote: [\[CITE\]](#) I'd like to see built-in script-daemon support.

What would be the benefits of this? [This SO question](#) has a number of answers that show how to run PHP in its own detached process, so I'm not sure what the additional benefits of this would be.

Haxalot - September 24, 2017, 2:28 pm

(Sep 24, 2017 01:59 PM)Yani Wrote: [\[CITE\]](#) Yeah I've read that question. But I just think it would be good to add. There wouldn't be a reason not to. PHP-FPM already runs as a daemon, so there's code already. Not to mention there's even an included webserver feature, and I'd say both would be equally useful.

I do agree that it would be a more convenient way to create PHP daemons. I'll add it to my feature list to make sure I don't forget about it, thanks.

Charger - September 24, 2017, 2:30 pm

Standard CONSISTENT function names. Like seriously, that's all it needs.

Haxalot - September 25, 2017, 2:01 am

(Sep 24, 2017 02:30 PM)Charger Wrote: [\[CITE\]](#) Standard CONSISTENT function names. Like seriously, that's all it needs.

Ain't gonna happen, which is why I listed it under the "do not suggest" list.

The problem with standardising function names (and by extension, parameter orders) is the backwards compatibility breakage it'll have. It will break practically all PHP applications, and so it will be a big reason not to upgrade. Such big breakages to the language were seen in PHP 5.3, where many did not bother to update their version of PHP for a very long time because of it. We also saw what happens when such large breaking changes are introduced by a mature language with the Python 2 and Python 3 ordeal.

A possible solution would be to introduce methods on primitive types, such as the following:

This would enable for a new, consistent API to be put in place, whilst completely maintaining backwards compatibility. But the problem with this is that PHP developers would then have to learn two APIs. With the old procedural API not going away any time soon (if ever), they will still be forced to use it when dealing with other developers' code.

So there isn't a particularly good solution to this problem. It is currently being ameliorated by IDEs and by having clear documentation in the PHP manual.

RumbleFrog - September 25, 2017, 11:33 am

A JIT compiler, heard rumors it might happen, let's hope so.

Haxalot - September 29, 2017, 3:52 am

(Sep 25, 2017 11:33 AM)RumbleFrog Wrote: [\[CITE\]](#) A JIT compiler, heard rumors it might happen, let's hope so.

Yep, that is being actively worked on. I'm not sure when we will see it, though. Attempting to build it with Clang or OS X fails, and it looks like some cleaning up WRT error handling still needs to be done. Maybe we'll see it in PHP 7.3, maybe not.

I've written a few snippets of code to show their run time performance with and without OPcache and JIT enabled (performed in my Centos 7 Vagrant VM, not under strict conditions, best time of 4 runs).

100 million empty loop iterations:

PHP Code

```
<?php
```

```
for ($i = 0; $i < 100000000; ++$i);
```

Results:PHP runtime without OPcache or JIT

Code \$ time sapi/cli/php test.php

real 0m0.740s

user 0m0.674s

sys 0m0.007s

PHP runtime with OPcache and without JIT

Code \$ time sapi/cli/php -dopcache.enable_cli=1 test.php

real 0m0.741s

user 0m0.680s

sys 0m0.003s

PHP runtime with OPcache and JIT

Code \$ time sapi/cli/php -dopcache.enable_cli=1 -dopcache.jit_buffer_size=1M test.php

real 0m0.284s

user 0m0.255s

sys 0m0.007s

Mandelbrot with 10,000 iterations:

PHP Code

<?php

```
for ($y = -39; $y < 39; ++$y) {
    for ($x = -39; $x < 39; ++$x) {
        iterate($x/40.0,$y/40.0);
        // echo !iterate($x/40.0,$y/40.0) ? '*' : ' ';
    }
    // echo "\n";
}
```

```
function iterate($x, $y)
{
    $cr = $y - 0.5;
    $ci = $x;
    $zr = $zi = 0.0;

    for ($i = 0; $i < 10000; ++$i) {
        $temp = $zr * $zi;
        $zr2 = $zr * $zr;
        $zi2 = $zi * $zi;
        $zr = $zr2 - $zi2 + $cr;
        $zi = $temp + $temp + $ci;

        if ($zi2 + $zr2 > 16) {
            return $i + 1;
        }
    }

    return 0;
}
```

Results:PHP runtime without OPcache or JIT

Code \$ time sapi/cli/php test.php

real 0m13.133s

user 0m12.096s

sys 0m0.022s

PHP runtime with OPcache and without JIT

Code \$ time sapi/cli/php -dopcache.enable_cli=1 test.php

real 0m4.761s

user 0m4.383s

sys 0m0.010s

PHP runtime with OPcache and JIT

Code \$ time sapi/cli/php -dopcache.enable_cli=1 -dopcache.jit_buffer_size=1M test.php

real 0m0.793s

user 0m0.728s

sys 0m0.003s

If you have any snippets of code you'd like to see run against PHP (with and without OPcache/JIT enabled), then feel free to post them below.

liumbiwe - September 29, 2017, 4:45 am

(Sep 29, 2017 03:52 AM)Haxalot Wrote: [\[CITE\]](#) If you have any snippets of code you'd like to see run against PHP (with and without OPcache/JIT enabled), then feel free to post them below.

Can you do some tests (JIT) on real applications like wordpress or maybe on frameworks like symfony/laravel on booting?

Or link them if there are already done.

I'm wondering if it is stable enough to handle this kind of apps.