www.phparch.com

May 2020
Volume 19 - Issue 5

php[architect]

Unsupervised Learning

Hands-On Machine Learning With PHP, Part Two

Decoupling Drupal From Its Frontend System to Use in an Existing Website

Passwordless Authentication

ALSO INSIDE

Education Station: Anatomy of a Web Response

Community Corner: York-Region-PHP User Group

History and Computing: Transcontinental Railroad

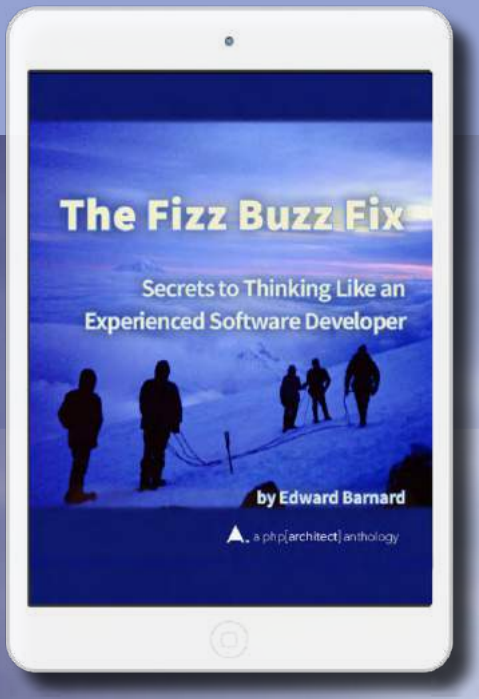
Security Corner: Request Replay Protection

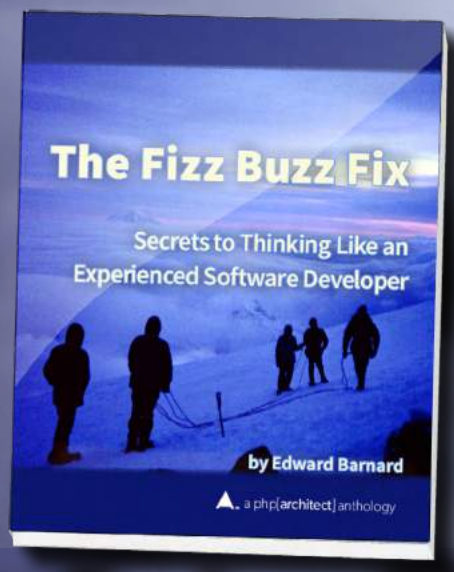
Puzzles: Factorials

The Workshop: Specification BDD with Phpspec

finally{}: What's in PHP Eight?

Free Sample Article





Tackle Any Coding Challenge With Confidence

Companies routinely incorporate coding challenges when screening and hiring new developers. This book teaches the skills and mental processes these challenges target. You won't just learn "how to learn," you'll learn how to think like a computer. These principles are the bedrock of computing and have withstood the test of time.

Coding challenges are problematic but routinely used to screen candidates for software development jobs. This book discusses the historical roots of why they select for a specific kind of programmer. If your next interview includes a coding exercise, this book can help you prepare.

Available in Print, Kindle Unlimited, and Kindle Lending Library

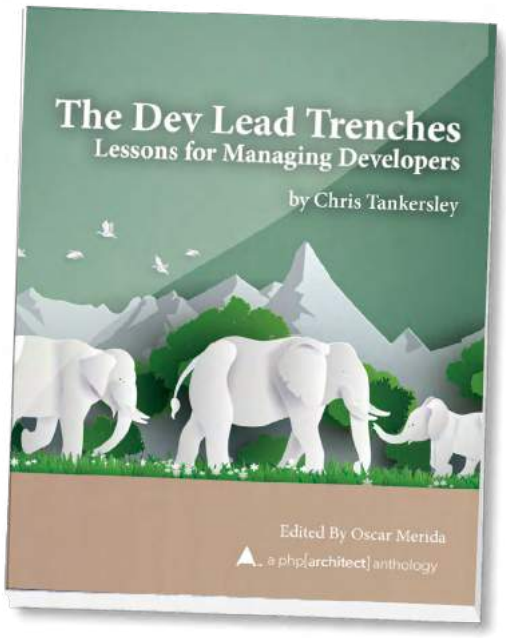
Order Your Copy

http://phpa.me/fizzbuzz-book

a php[architect] anthology

You're the Team Lead—Now What?

Whether you're a seasoned lead developer or have just been "promoted" to the role, this collection can help you nurture an expert programming team within your organization.



Get the Most Out of Your Developers

After reading this book, you'll understand what processes work for managing the tasks needed to turn a new feature or bug into deployable code. But success is more than just slinging code when you're in charge, and this book covers project management and people skills you'll need to hone.

This book collects almost two-years worth of writings based on Chris Tankersley's experience leading development teams.

Available in Print, PDF, EPUB, and Mobi.

Order Your Copy

http://phpa.me/devlead-book

What's in PHP Eight?

Eli White

While much of the world shuts down, the PHP core developers have been hard at work preparing for the release of PHP 8.0 at the end of this year! The feature freeze is in just a few months (July 28th), so this is the exciting time when there is a push to get various features into this momentous release! Let's take a look at a few of the bigger things currently planned for PHP 8.0

$object::class

The `::class` construct has existed since PHP 5.5 and allows you to get the name of a class as a string. However, it could only be called directly on the class itself. It makes sense to use this syntax on objects as well. So that is being implemented! This functionally equivalent to calling `get_class()` on the object, but without needing the function wrapper.

Static Return Type

Hopefully you are familiar with the `static` keyword—technically special class name—which allows for late static binding in PHP. I have been known to use it perhaps a bit too much. Essentially it references the class a method was called on, even when that method was inherited. Consider the code lin Listing 1.

The code echoes `Oscar` which is what you'd expect and want. Using `self` would instead have output `Eli`. The new `static` return type now does the same thing, evaluating the name of the class on resolution. For example, if we added the following to the above Author class:

```
public function fluent(): static {
    // Do stuff
    return $this;
}
```

Then a call to he `Editor` class' `fluent()` method acts as having a return type of `Editor`. And a call to `Author`'s is `Author`.

Union Types

When declaring a variable, parameter, or return type in PHP, you have always had to declare a single type. This behavior was updated recently to allow for a special either/or union of a type *or* null value using the `?type` syntax. However, a strong power of PHP has been it's type flexibility. Being able to make a function that perhaps returns `NULL`, a `false`, or a `string`, is handy sometimes. The type system is being updated to allow for that. So anywhere you can currently place types, you'll be able to union multiple possible types together with the `|` syntax such as:

Listing 1

```
1. <?php
2.
3. class Author
4. {
5.     public function who() {
6.         echo "Eli\n";
7.     }
8.
9.     public function name() {
10.        static::who();
11.    }
12. }
13.
14. class Editor extends Author
15. {
16.    public function who() {
17.        echo "Oscar\n";
18.    }
19. }
20.
21. $e = new Editor();
22. $e->name();
```

```
function weird(int|float $number): null|boolean|string {
    /* Do things */
}
```

Along with this, a new pseudo-type of `false` is being created. It allows a function that returns a value or false, as many older PHP functions do, to declare their intended usage. Similarly, while you can still use the `?` notation to denote a "type or null" situation, you can also now specify that directly as I did in the example above.

str_contains()

For decades, PHP developers have repurposed `strpos()` and `strstr()` to check if a string exists inside of another one. However, the way you use these is often confusing to newer developers (especially with the need to check `false` versus `0`. PHP 8.0 fixes that by adding a new function that returns a boolean to indicate whether one string exists inside of another:

```
str_contains ( string $haystack , string $needle ) : bool
```

JIT (Just in Time Compiler)

PHP 7 brought some substantial performance gains, which were initially inspired by some work on creating a JIT for PHP. While gains were found during the process, the JIT itself wasn't released because it was difficult to maintain and wasn't showing any positive speed improvements for web applications at the time.

Well, it's ready to be released now. It's been updated and augmented. Meanwhile, the nature of many web apps has changed to now make it even more relevant. It's implemented as a part of OPCache. However, you can toggle it on or off independently.

Currently, it doesn't appear to make web applications, such as WordPress, any faster; however, in CPU intensive workloads within PHP, it is showing up to a 2x speed increase.

Summary

This overview has been just a personal highlight reel of things coming to PHP 8. There are lots of other enhancements and fixes coming as well, with three months for new features to still make it in! As we approach the end of 2020 and into 2021, it's going to be an exciting time for PHP!

Eli White is a Conference Chair for php[architect] and Vice President of One for All Events, LLC. He doesn't have any personal upgrades planned for himself at the moment. Maybe blue hair? @EliW

Borrowed this magazine?

Get php[**architect**] delivered to your doorstep or digitally every month!

Each issue of php[**architect**] magazine focuses on an important topic that PHP developers face every day.

We cover topics such as frameworks, security, ecommerce, databases, scalability, migration, API integration, devops, cloud services, business development, content management systems, and the PHP community.

Digital and Print+Digital Subscriptions Starting at $49/Year

http://phpa.me/mag_subscribe

php[**architect**]