zend
by Perforce

The Shifting Landscape of PHP Report

www.zend.com

Introduction

PHP Background

PHP, the open source programming language originally created for making simple web pages, has been maturing over the past few years. According to W3Tech, 80% of the world's websites use PHP. This trend has been holding steady, and thanks to innovations like PHP 7, which brought tremendous performance gains to PHP, the language is thriving. The future is promising for PHP, with PHP 7.4 and PHP 8 on the horizon, along with exciting features like FFI and JIT.

Executive Summary

What is the current state of PHP, and where is it heading?

The landscape has already started to shift, with new tools and architectures being introduced that have huge implications on application development as we know it. To prepare PHP developers for continued changes in the market, the Zend team conducted a survey at the end of March 2019.

We wanted to collect these insights in order to share knowledge and useful information with the PHP community.

Survey Methods

Out of the 539 total survey respondents, more than half were software developers (55.3%) while a quarter were managers or team leads. Half (50.3%) were on a team of between two and five developers. We do not know what ratio of of participants are part of an enterprise versus a smaller or startup organization.

The majority of participants were using PHP version 7.2 or 7.3 (56.6%), with only 2.6% using version 5.5 or earlier.

Current State of PHP

Developer Use of PHP

We first offered questions relating to developers' use of PHP in their applications, asking, "What are you using PHP for?" Respondents could select multiple options from the given list. Not surprisingly, most are using the language for web applications, followed by Application Programming Interfaces (APIs) and services.

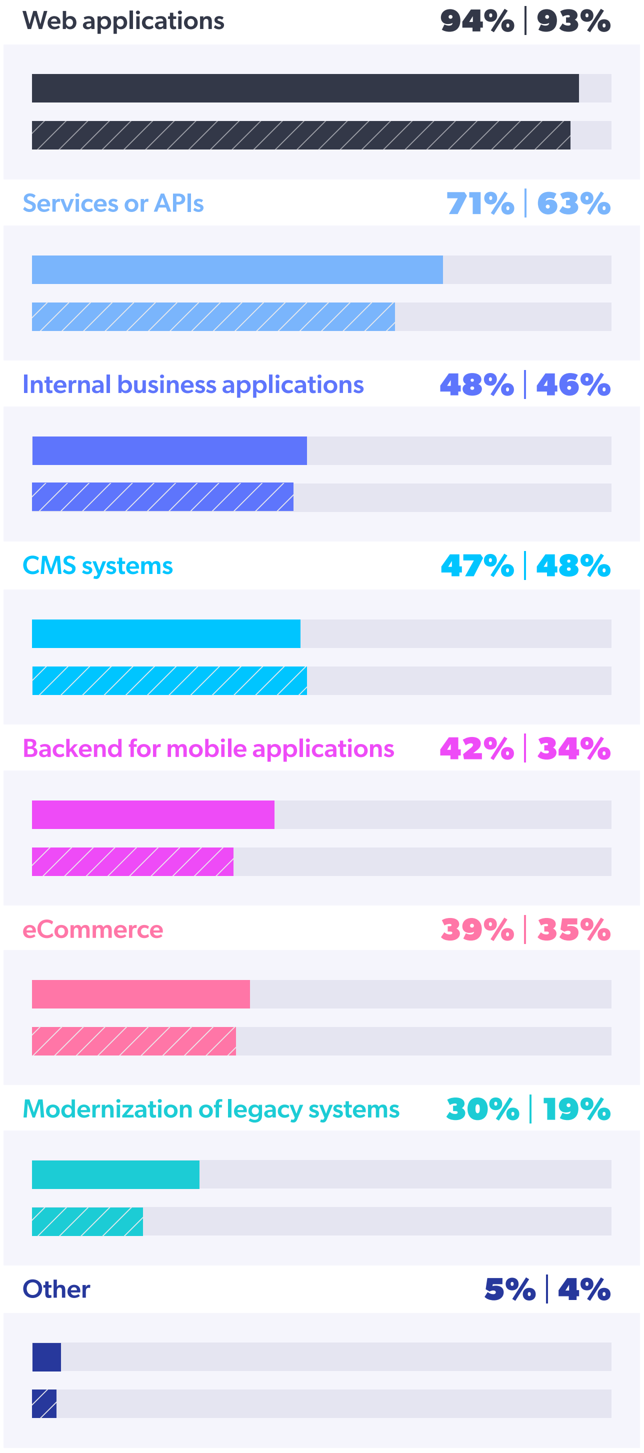
Compared to a similar survey Zend created in 2016, the use of PHP for APIs and services has increased by over 7%. We predict this will continue to rise to a point where APIs surpass web applications in terms of development with PHP. Over time, we will see less traditional monolithic web applications as developers move toward services-oriented architectures and microservices—but more on that later.

Another change from 2016 is the use of PHP in mobile application back-ends, which increased from 34.4% to 41.9%. As our world becomes increasingly mobile, our development becomes API-first to serve not only mobile devices, but provide a central hub for information that apps, front-end web applications, wearables, and IoT devices can connect to.

Application programming interface (API): Processes or functions for communication among various program components.

Microservices: Loosely-coupled fine-grained services that allow developers to create, manage, update, test, and scale different features of an application separately.

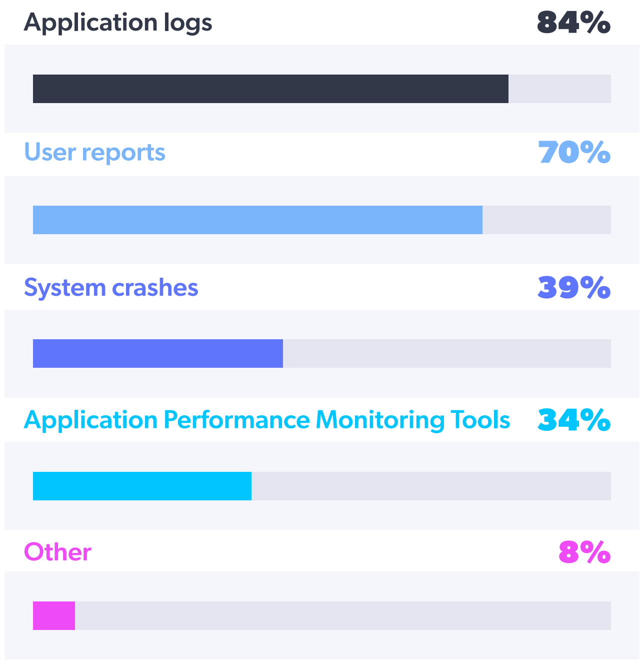
WHAT ARE YOU USING PHP FOR?



PHP Performance and Security

Next, we wanted to learn about how developers are identifying production issues and what tools they are using, if any, to monitor application performance. When asked, "How do you discover issues in your product applications?" we discovered that most are identifying problems via application logs (84%), followed by user reports (70.1%).

HOW DO YOU DISCOVER ISSUES IN YOUR PRODUCTION APPLICATIONS?



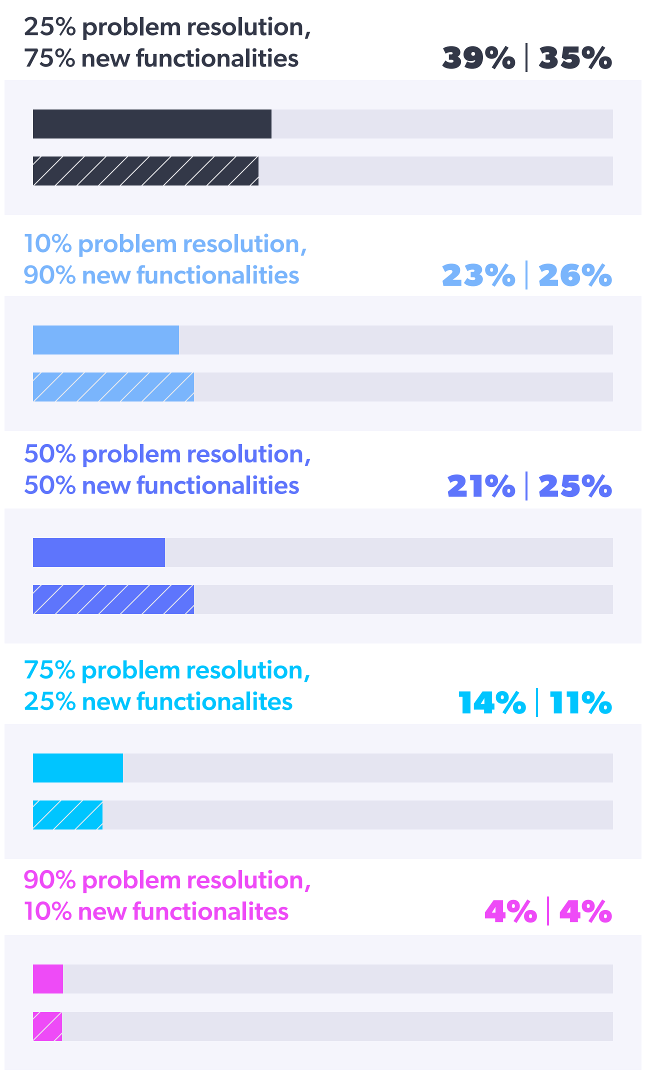
What is startling here is that nearly 40% of developers aren't uncovering performance issues until the system crashes.

Additionally, when asked what Application Performance Management (APM) tool developers are using in production, more than half (56.4%) indicated they aren't using one at all. Paired with the fact that a large number of developers find out about issues only once their system crashes, there is a clear lack of performance supervision during the development process. APM tools are designed to spot issues before they become severe and reduce time spent debugging.

With all this in mind, it's no surprise that developers are spending more than a quarter of their time investigating and solving problems. In our 2016 survey, the majority also indicated that they spend 25% of their time in problem resolution. We find it interesting that this number hasn't decreased more significantly with the advancement of tools, automation, and stable frameworks. This could be attributed to developers resisting change and learning new solutions, or the DIY nature of OSS.

Application Performance Management (APM): The monitoring of a software application's performance.

HOW MUCH OF YOUR TIME IS SPENT ON MAINTENANCE AND PRODUCTION BUG/ISSUES RESOLUTION VS. DEVELOPING NEW FUNCTIONALITIES?



Although small, over 4% are spending 90% of their time resolving problems, and only 10% on building new functionalities. While it's expected that some time will be spent identifying issues, unless it's a developer's main responsibility, it should not be taking up a majority of their workload.

Nearly two-thirds (65%) of developers indicated they are responsible for their application security. Depending on the size of an organization and its IT team, a developer may take on more or less of this important element of their development; however, there is a level of security inherently incumbent to developers.

What this number really tells us is that more and more developers are aware of security and its implications, and because of that they are becoming more cautious. While there are tooling and services for security that offload some of the work for developers, there are parts that they will always need to handle, such as input escaping and validation, CSRF and XSS prevention, and SQL injection—which one would expect to be a problem of the past with prepared statements and the advancement of database abstraction libraries, but surprisingly still tops the OWASP top 10 list.

The PHP Roadmap

WHO IS THE PERSON MOST RESPONSIBLE AND ACCOUNTABLE FOR SECURITY IN YOUR SOFTWARE DEVELOPMENT PROCESS?

Individual developers	45%
Development managers	21%
IT operations	13%
It's not clear who is responsible	11%
CSO	7%
Cloud/hosting provider	3%
External security consultants	1%

Frameworks make the task of protecting one's application easier, but efforts to raise awareness within organizations and communities is one of the most effective ways to increase developers' skillsets on security.

New Tools and Technology

As we mentioned, more developers are migrating away from monolithic applications to microservices architectures. In fact, 65% of survey respondents indicated that they are either already using, currently implementing, or planning to implement microservices in new or existing applications.

ARE YOU PLANNING TO MIGRATE TOWARDS MICROSERVICES ARCHITECTURE?

No	35%
Yes - For new applications only	24%
Yes - But have not determined when	17%
We're already using microservices	14%
Yes - Within a year	10%

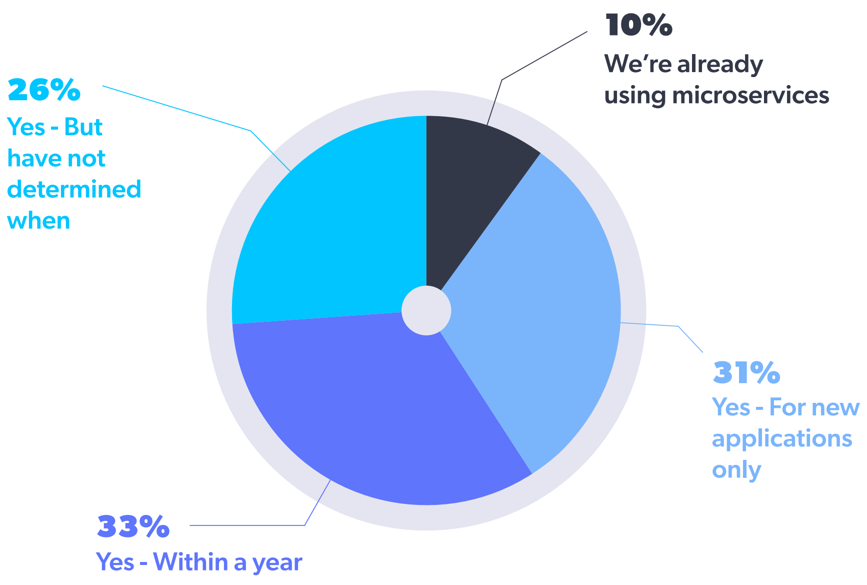
The main driver to microservices is that developers aren't restricted to one language or platform. The end result is a collection of loosely-coupled services can talk to each other with a lightweight protocol. Different teams working on different features are able to write their code the way they want. All they need to know are the interfaces of the services they are consuming, and they can publish the interfaces of the services that they are developing.

However, it's important to be aware of the potential drawbacks. Microservices isn't necessarily the best option for every application, and implementing this architecture introduces new levels of complexity. While the best approach depends on the particular situation of the implementer, we generally recommend developing an application as a monolith, then breaking it up into microservices. This is especially beneficial for startups and smaller organizations that may lack the resources and manpower for such an undertaking and may not know yet which features and functionalities will need to be separated, and may fall into the easy trap of overengineering early in the process.

Async is another programming paradigm gaining traction. We asked, "Are you currently using, or do you plan to use PHP in an asynchronous manner (e.g., Swoole)?"

Asynchronization (async): Form of programming where an application element runs separately from the primary application.

ARE YOU CURRENTLY USING OR PLANNING TO USE PHP IN AN ASYNCHRONOUS MANNER (E.G. SWOOLE)?



Close to half (41.3%) are either currently implementing or looking into implementing asynchronous processing. It's still in early stages with PHP but can provide impressive performance gains (four to ten times), requiring less resources and making it an attractive option. However, it's not an easy switch to make, as it is somewhat different than the PHP code we are used to, and converting existing applications to leverage asynchronous processing requires a fair bit of refactoring.

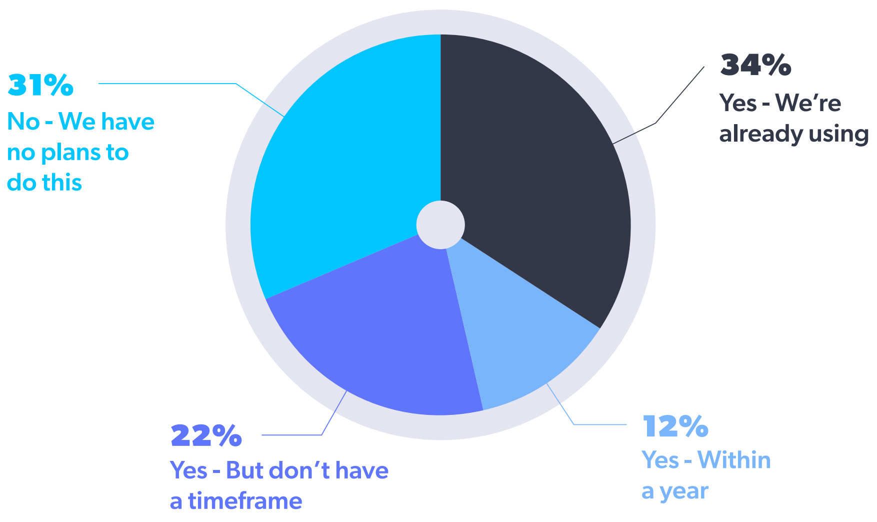
Finally, containerization adoption is also steadily growing, with 34% indicating they are already leveraging containers, and 34.6% planning to use them.

As the lowest barrier of entry, the numbers on containers aren't shocking. One thing to note, however, is that the advantage of moving into a containerized environment comes with microservices. Containerizing a monolith application doesn't offer as much benefit.

These three advancements can all be costly and difficult to get right. It's easy to get drawn to the newest, shiniest product, but it's important to consider the costs and drawbacks.

Containers: Mechanism for isolating one application or feature from another.

ARE YOU CURRENTLY USING OR PLANNING TO USE ANY CONTAINERIZATION TECHNOLOGIES?



Conclusion

While containers, async, and microservices can all be leveraged separately, there are some interesting synergies when they are composed together, so it's safe to say the industry is going to continue moving in this direction in the coming years.

In the current environment, developers need to stay up-to-date on where things are moving, and whether new tools, such as microservices, are applicable in their daily work. Only focusing attention at development won't cut it; they need to understand how their programs run as well as the tools and implications. Skillsets are more important than ever, so developers must keep them in check.

In technology, there are cycles with new innovation, and we typically don't notice the disruption until it actually happens. You don't know what the disruptor will be before it's everywhere. We are currently in the beginning of one of these cycles, and developers should expect and prepare for major movement over the next three to five years. The changes on the horizon will redefine a lot of what we do.

ZEND SERVER IS YOUR ENTERPRISE PHP SOLUTION

For those needing support with their PHP development and production, Zend Server is a tool that allows application developers speed up deploys. Our strategy surrounding the Zend Server product will focus on the evolving PHP landscape and implementation of new solutions.

Not only does our platform assist with the migration toward the microservices and container architecture, we have the services and consultants to support you as your development continues to evolve.

Learn more by visiting us online at https://www.zend.com.