

# PHPの20年とこれから

20 years of PHP and beyond



pixiv Inc.  
USAMI Kenta

Docomo R&D OPEN LAB ODAIBA  
2023-08-26 Learn Languages

# お前誰よ



- うさみけんた (@tadsan) / Zonu.EXE / にゃんだーすわん
- ピクシブ株式会社 pixiv事業本部 Webエンジニアリングチーム PHPer
  - 2012年末から現職でPHPを書いているWebプログラマ
- Emacs PHP Modeを開発しています (2017年-)
- プログラミング言語にちょっとこだわりのある素人 (spcamp2010)

# emacs-php

emacs-php

Q Type / to search

Overview

Repositories40

Projects

Packages

Teams1

People9

Settings



Friends of Emacs-PHP development

Join us!

6 followers <?php

Pinned

Customize pins

php-ts-modePublic

A Tree-sitter based major mode for editing PHP codes

Emacs Lisp53

php-modePublic

A powerful and flexible Emacs major mode for editing PHP scripts

Emacs Lisp566117

phpactor.elPublic

Interface to Phpactor (an intelligent code-completion and refactoring tool for PHP)

Emacs Lisp4011

phpstan.elPublic

Interface to PHPStan (PHP static analyzer)

Emacs Lisp2412



<?php



PHPって  
どんな言語？

とにかく毀誉褒貶が  
激しい言語ランタイム

# Webトラフィック



# W3Techs 統計

“

Technologies

Content Management

Server-side Languages

Client-side Languages

JavaScript Libraries

CSS Frameworks

Web Servers

Web Panels

Operating Systems

Web Hosting

Data Centers

Reverse Proxies

DNS Servers

Email Servers

SSL Certificate Authorities

Content Delivery

Traffic Analysis Tools

Advertising Networks

Tag Managers

Social Widgets

Site Elements

Structured Data

Markup Languages

Character Encodings

Image File Formats

Top Level Domains

Server Locations

Content Languages

Trends

History

Market

Top Site Usage

Market Position

Performance

Technologies > Server-side Languages

Usage statistics of server-side programming languages for websites

Request an extensive server-side programming languages market report.

[Learn more](#)

This diagram shows the percentages of websites using various server-side programming languages. See [technologies overview](#) for explanations on the methodologies used in the surveys. Our reports are updated daily.

How to read the diagram:

PHP is used by 77.2% of all the websites whose server-side programming language we know.

|              |       |
|--------------|-------|
| PHP          | 77.2% |
| ASP.NET      | 6.9%  |
| Ruby         | 5.4%  |
| Java         | 4.7%  |
| Scala        | 2.9%  |
| JavaScript   | 2.5%  |
| static files | 1.9%  |
| Python       | 1.5%  |
| ColdFusion   | 0.3%  |
| Perl         | 0.2%  |
| Erlang       | 0.1%  |

W3Techs.com, 26 August 2023

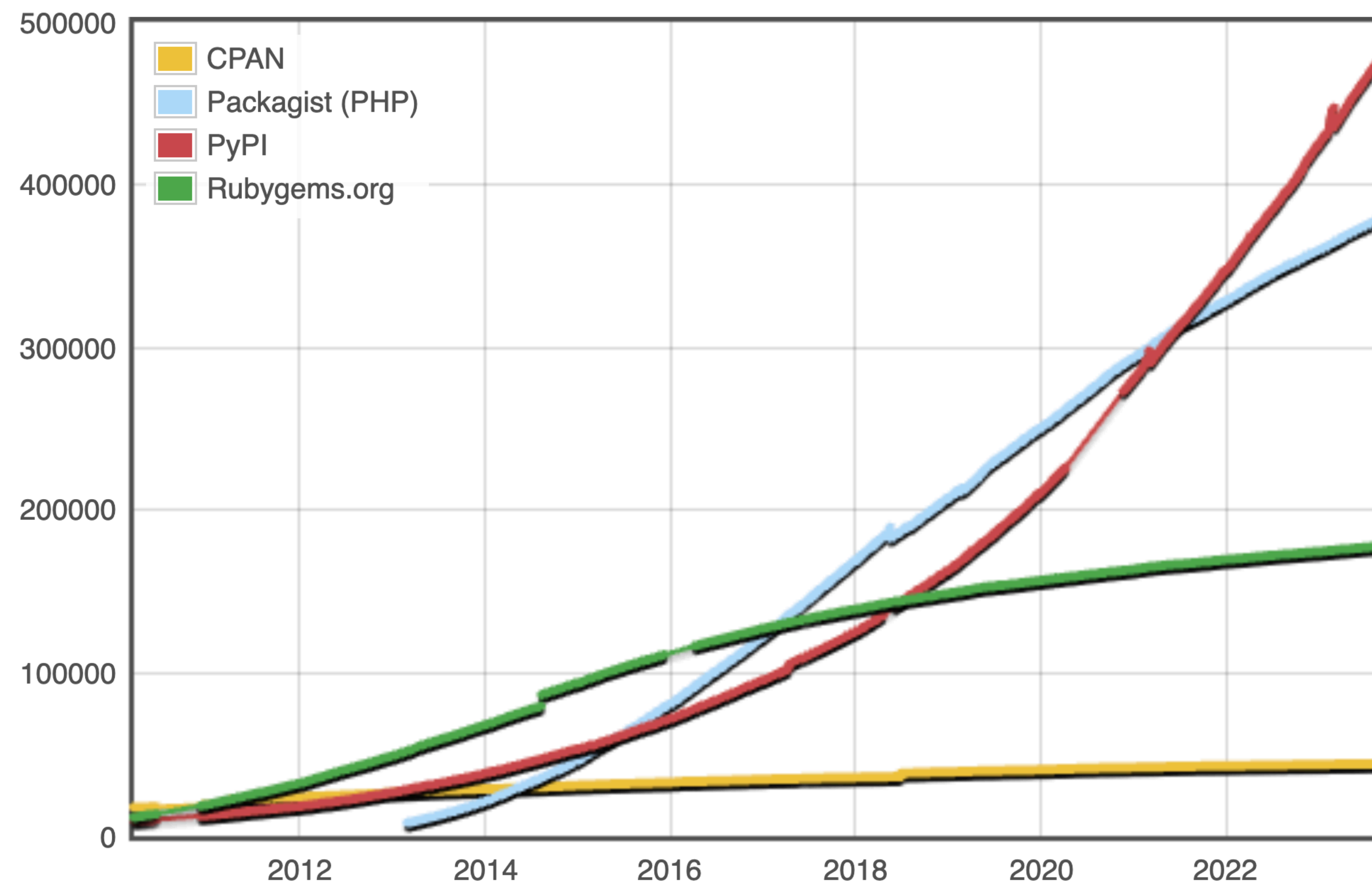
Percentages of websites using various server-side programming languages  
Note: a website may use more than one server-side programming language

The following server-side programming languages are used by less than 0.1% of the websites

”

# パッケージ数

## Module Counts



PHPはどこで  
使えるのか



# PHPは汎用プログラミング言語

- HTTP = Webサービス



# PHPは汎用プログラミング言語

- HTTP = Webサービス
- それ以外の任意のあらゆること（得意とは言っていない）

# PHPは汎用プログラミング言語

- HTTP = Webサービス
- それ以外の任意のあらゆること（得意とは言っていない）
  - CLI
  - GUIもいちおう…（得意とは言っていない）

# みなさんの想像する PHPとは



```
<!DOCTYPE html>
<html><head><title>Hello, PHP!</title></head>
<p>
  <?php if (date('H') < 19): ?>
    こんにちは
  <?php else: ?>
   こんばんは
  <?php endif; ?>
</p>
<p>今は<?= htmlspecialchars(date('H')) ?>時です</p>
```

```
<?php require 'bootstrap.php';

$db = mysqli_connect();
$stmt = mysqli_prepare($db, 'SELECT * FROM books');
mysqli_stmt_execute($stmt);
print '<h1>本の一覧</h1>';
print '<ul>';
foreach ($stmt as $s) {
    print '<li>'; print $s->name; print '</li>';
}
print '</ul>';
```

```
<?php declare(strict_types=1);

namespace Foo\Http\Controller;

class BooksController extends BaseController {
    public function index() {
        $books = Books::getAll();

        $this->render(compact("books"));
    }
}
```



PHPの原作者

ラスマスかく語りき

# 2007年 MySQL Conference

*“We have things like protected properties.  
We have abstract methods. We have all this  
stuff that your computer science teacher told  
you you should be using. I don't care about this crap at all.”*

*–Rasmus Lerdorf*

[https://en.wikiquote.org/wiki/Rasmus\\_Lerdorf](https://en.wikiquote.org/wiki/Rasmus_Lerdorf)



# 2007年 MySQL Conference

“PHPには`protected`プロパティも  
抽象メソッドもありますよ。計算機科学の  
教授が「使え」と言ってるものは全部。  
そんなことはクソ興味ないですけど”

–Rasmus Lerdorf (*tadsan*による超訳)

変数に\$があって  
Perlっぽくて

C言語っぽい  
関数と  
->があって



# 雰囲気Javaっぽい オブジェクト指向で

# PHPは見る人の心を映す



改めて2003年



# PHPの歴史 (LLイベント以前)

- 1994: PHP Tools 1.0 → SSI風構文のテンプレートエンジン
- 1996: PHP/FI 2.0 → 汎用言語として使用できるように
- 1998: PHP 3.0 → クラス構文のサポート
- 1999: PHP 4.0 → Zend Engine (インタプリタVM化)
- 2003: PHP 5.0(開発中) → オブジェクト指向構文の刷新





## PHP の歩み



1995

Rasmus Lerdorf により PHP/FI 開発

PHP/FI

1996

PHP/FI 日本語 Web ページ, 日本語 ML  
PHP/FI 日本語マニュアル, PHP/FI 日本語パッチ

2

1997

PHP3 開発 (新開発体制へ移行)

PHP3

1998

PHP3 日本語マニュアル作成プロジェクト  
PHP3 日本語パッチ

1999

PHP 国際化プロジェクト開始  
「PHP 徹底攻略」出版



2000

日本 PHP ユーザ会設立  
PHP4 リリース: ZendEngine による性能向上

PHP4

2001

PHP 4.0.6 : 日本語対応マージ

2002

PHP 4.1 : メソッド毎の入力取得, マルチバイト正規表現  
PHP 4.2 : register\_global=Off がデフォルトに

2003

PHP 4.3 : マルチバイト対応強化 (CJK、スクリプト文字コード)  
(PHP 5 : オブジェクト機能強化: ZE2)

2004

PHP5

Domains  
(netcraft)  
15,000

50,000

150,000

2,000,000

全ドメインの 30%  
Apache の 50%

12,000,000

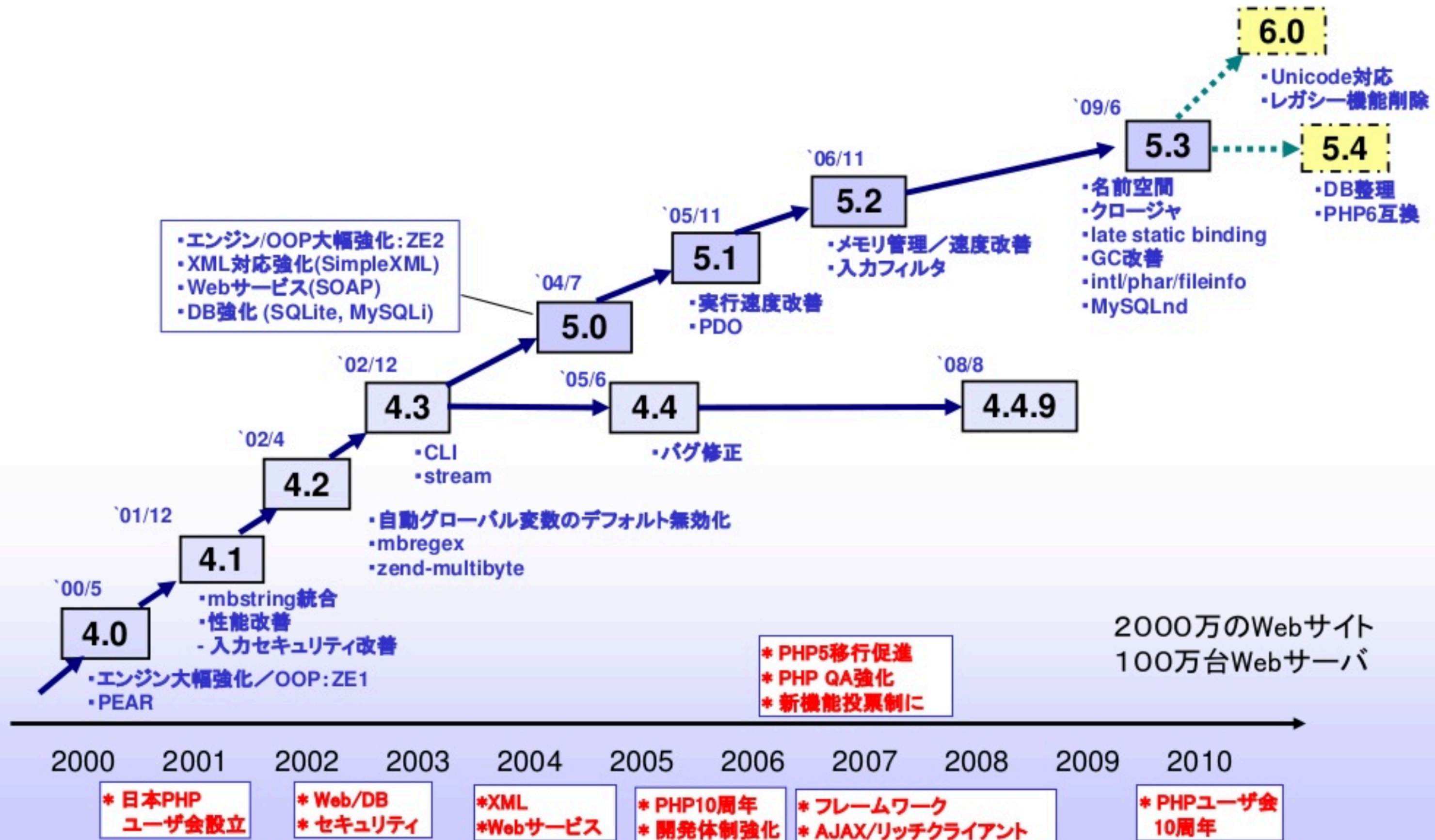
– PHPの今とこれから

<http://ll.jus.or.jp/2003/document/p1-php.pdf>





# PHPの歩み



– PHPの今とこれから2009

<https://www.slideshare.net/hirokawa/php2009>



## PHP 6.0

- PHP 6は遅延中： 2009/5に開発者会議開催
- 国際化／Unicode化
  - Unicodeネイティブ対応(ICU): 常にオン
  - 国際化機能:intlエクステンション (PHP 5.3と共通化)
  - 書き直しの範囲大:パーサ(re2c)、DB拡張(PDO)など
  - DB: 接続文字コード、テーブル文字コード、カラム文字コードの区別
- レガシーコード削除
  - register\_globals, magic\_\*, safe\_mode
- 論議中:
  - 識別子で大文字／小文字区別 PHP 6.1 (PHP 5.4で警告)
  - Traits 継承の柔軟化

– PHPの今とこれから2009

<https://www.slideshare.net/hirokawa/php2009>



## 開発中の PHP 6、UTF-16 化に失敗。開発ブランチも 5.3 系に巻き戻し

ストーリー by reo 2010年03月19日 12時30分 決断には勇気を要したことだろう 部門より

ある Anonymous Coward 曰く、

PHP の次期メジャーバージョンと見られている PHP6 では、内部的には文字列をすべて UTF-16 で処理するという方針が決定していたのだが、これが頓挫した模様 ([マイコミジャーナルの記事](#))。

PHP 開発者である Johannes Schlüter 氏による 2010/3/12 付けのブログ記事、"[Future of PHP 6](#)" によれば、数力月前から PHP のコア開発者の多くから「PHP エンジン内部を Unicode 化するというアプローチは正しくないのでは、最初からやり直したほうがよいのでは」という議論が行われていたらしい。

「処理系内部ではすべての文字を Unicode で処理する」というアプローチは Java や Ruby、Python、Perl などですでに採用されているのだが、PHP の開発者らの結論は「プログラムにおいてすべての入出力時に変換処理を行うのはパフォーマンスの点でよろしくなく、実装も複雑になり、後方互換性もなくなる。いっぽうで多くのユーザーが受ける恩恵は非常に小さい」とのことで、とりあえずは現在行われていた PHP エンジンの UTF-16 化はすべて白紙に戻されるようだ。

77 コメント ◆ ◆php ◆デベロッパー



– 開発中の PHP 6、UTF-16 化に失敗。開発ブランチも 5.3 系に巻き戻し  
<https://srad.jp/story/10/03/19/0136250/>

# PHPと文字列

- string型 = バイト列（任意のバイナリが格納できる）
  - Ruby 1.8、Python 2.xと同等
- PHP 6.xでICUと統合してUCS正規化しようとした
  - 規定路線だったので、PHP 5.2(2006年)にbinaryキャストが追加
  - 当時期待されていたパフォーマンス要件を満たせず2010年に断念
- 文字列以外の機能は5.3(2009年)、5.4(2012年)でリリース

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 12:15-13:30 昼休み |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 13:30-15:00     | <p><b>パネルディスカッション: 「LLとオブジェクト指向について」</b></p> <p><b>司会:</b> 小山哲志(日本UNIXユーザ会)</p> <p><b>オブザーバー:</b> まつもとゆきひろ(ネットワーク応用通信研究所)</p> <p><b>Perl:</b> 宮川達彦(エッジ)</p> <p><b>PHP:</b> 藤本真樹(TUNEBiZ)</p> <p><b>Python:</b> 磯蘭水(日本Pythonユーザ会/青山学院大学)</p> <p><b>Ruby:</b> 中田伸悦</p> <p>今回対象としている4言語は、オブジェクト指向に関して考え方も実装している機能もそれぞれ異なります。</p> <p>各言語の代表者の方々にその言語のオブジェクト指向機能について解説していただき、各言語の特徴を明確にするのがこのセッションの目的です。</p> <p>後半は一般参加者も交えてのディスカッションとなります。オブジェクト指向に一言ある方の積極的な参加をお願いします。</p> <p><a href="#">(当日のログ)</a></p> |

– Lightweight Language Saturday プログラム  
<http://ll.jus.or.jp/2003/program.html>





Lightweight Language Saturday

**PHPにおけるオブジェクト指向  
「的」  
機能の特徴**

藤本 真樹  
fujimoto@php.net

有限会社チューンビズ  
104 0045  
東京都中央区築地2-1-14トミービル5F  
03 5148 0113 / fax 03 5148 0114  
<http://tunebiz.net/>  
fujimoto@tunebiz.net

**TUNEbiz.Net**  
tuning business & network

August, 9, 2003

– PHPにおけるオブジェクト指向「的」機能の特徴  
<http://ll.jus.or.jp/2003/document/p2-php.pdf>





## History

### オブジェクト指向的歴史

- PHP/FI
  - なし
- PHP3
  - 最小限のサポート
    - クラス宣言/単一継承/コンストラクタ
- PHP4
  - 内部構造の改善
  - リファレンスサポート
- PHP5(coming soon)
  - アクセス制御(private, protected)/デストラクタ/コピーコンストラクタ/etc...

August, 9, 2003

– PHPにおけるオブジェクト指向「的」機能の特徴  
<http://ll.jus.or.jp/2003/document/p2-php.pdf>

# PHPのオブジェクト指向機能

- PHP 5.0
  - 可視性 (public/protected/private)
  - abstract/interfaceが導入
  - オブジェクトが参照型に（それまではCoW）
- PHP 5.4: traitが導入
- PHP 5.6: 定数式（それまではリテラルしか記述できなかった）



# PHPのオブジェクト指向機能

- PHP 7.4: 型付きプロパティ
- PHP 8.0: コンストラクタプロモーション (PHP 8.2)
- PHP 8.1: readonlyプロパティ
- PHP 8.2: trait定数宣言
- PHP 8.3: 型付き定数宣言

```
<?php
```

```
class Book {
```

```
    /**
```

```
     * @var string
```

```
     * @readonly
```

```
    */
```

```
    private $name;
```

```
    /** @param string $name */
```

```
    public function __construct($name) {
```

```
        $this->name = $name;
```

```
    ...
```

```
<?php
```

```
class Book {
```

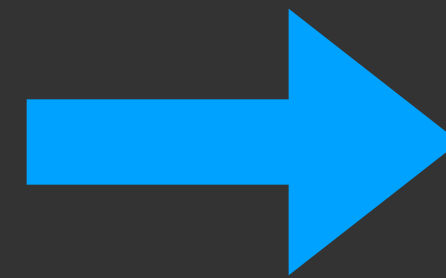
```
    public function __construct(
```

```
        public readonly string $name,
```

```
    ) {
```

```
    }
```

```
    ...
```



# PHPの歴史 (LLイベント以後)

- 2004: PHP 5.0
- 2005: PHP 5.1
- 2006: PHP 5.2 (PHP 6との互換性を見越したsyntax追加)
- 2009: PHP 5.3 (PHP 6予定の機能の一部を先行マージ)
- 2012: PHP 5.4 (PHP 6開発断念、文字列以外の機能をマージ)



# PHPの開発体制

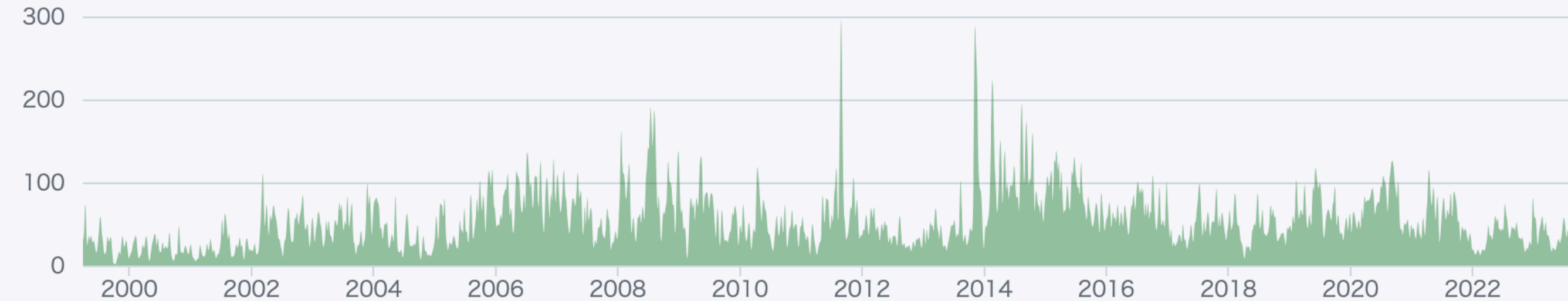
- Request for Comments <https://wiki.php.net/rfc>
  - 言語機能の改廃提案と投票プロセス、リリース周期が明文化
- 機能提案する人は仕様をまとめてMLで議論、実装も責任を持って用意する
- PHP原作者のRasmus Lerdorfは開発者としての1票しか行使しない
- PHP 7.0(2015年)以降、11月末～12月初頭にリリースが定着
  - 品質保障のためのfeature freeze、RC版なども定められた

# PHPは誰が開発しているのか

Apr 4, 1999 – Aug 26, 2023

Contributions: Commits ▾

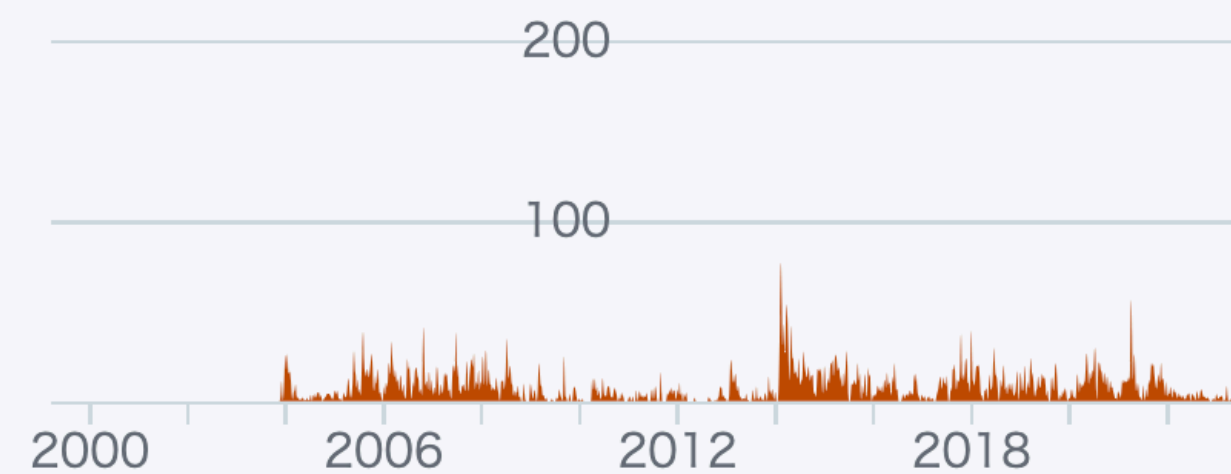
Contributions to master, excluding merge commits and bot accounts



**dstogov**

8,207 commits 950,566 ++ 587,128 --

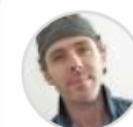
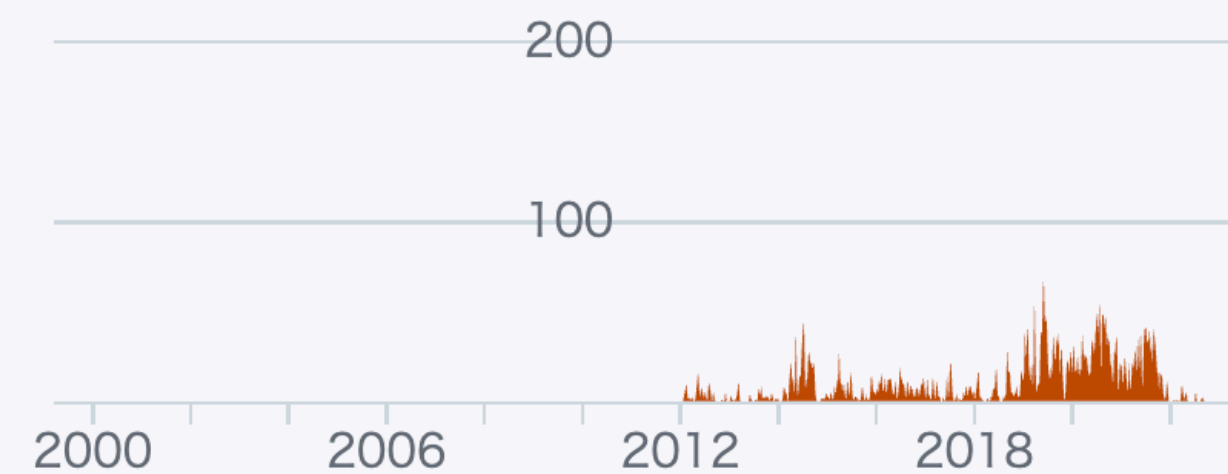
#1



**nikic**

5,832 commits 532,032 ++ 773,423 --

#2



**weltling**

4,917 commits 3,415,877 ++ 1,934,631 --

#3



**helly25**

4,491 commits 229,926 ++ 104,171 --

#4



Tuesday, 4 May 2021

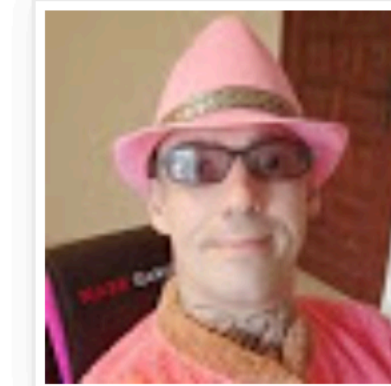
## Avoiding Busses



Fig 1. A Bus

It's always been the case that there are certain parts of PHP source code that only a few people understand. The Karma system used to help us determine where a contributor could commit code in the source tree; If you had /Zend karma, you had a clue about Zend. Among those people with /Zend karma, some people understood more than others.

### About Me



 Joe Watkins

[View my complete profile](#)

### Blog archive

▼ 2021 (2)

▼ May (2)

[Avoiding Busses](#)

[Worthiness](#)

► 2020 (2)

► 2019 (12)

► 2018 (4)

► 2017 (1)

► 2016 (7)

► 2015 (11)

► 2014 (17)





# インフィニットループは PHP の継続的な発展を目指す The PHP Foundation に寄付をしました

## TL;DR

この記事では、以下の内容について述べています。

- PHP は主要開発者が抜け、今後のために Foundation を作りました
- 弊社は支援を行いました
- 継続的な PHP の成長のためにみなさんの会社でも寄付を考えていただけませんか

## 経緯など

日本時間で 11 月 23 日の午前 2 時頃、JetBrains の The PhpStorm Blog にて、以下 2 つの [PHP にとって重大な発表](#)が行われました。

- [Nikita Popov \(nikic\)さん](#)が 12 月から活動の主軸を PHP から LLVM に移すこと
- これを契機としての The PHP Foundation の設立

この 2 つの事件により、PHP はプログラミング言語として大きな転機を迎えています。

弊社インフィニットループはかねてより PHP を業務で使用してきており、PHP の言語としての発展と継続にその利益をある程度依存しています。道義的な側面でもある種の投資的な側面でも、元々何らかの形でコミュニティへ利益を還元すべきという認識はありつつ、PHP という言語自体へ適切な形で還元を行う方法が分からないでいました。The PHP Foundation の設立により、今回この問題が解消したわけで

りオンラインにて開催！

15:15

Sponsor Session : ChatAI との上手な向き合い方

講師：千葉 龍弥 (株式会社インフィニットループ プログラマー)

[@iloop\\_sapporo](#)

[#ldev23sec](#) [#local\\_do](#)



1 4



## 最新の記事

[【Unity】BlendShapeを倍速化！GPU処理を自分で書いて、Unity公式BlendShapeのパフォーマンスに勝ってみよう！](#)

[PHPカンファレンス沖縄 2023 に IL から 2 名採択されました！](#)

[【Unity】AnimatorやPlayableGraphからも卒業！？スキニング・スケルタルアニメーションの](#)

# Interview with Core Developers

Published on May 06, 2022 by [Sebastian Bergmann](#) Interview

The initial group of sponsored developers has now been "at it" for a month, and we thought that you might be interested in who they are and what they work on. I got the opportunity to (virtually) sit down with Derick Rethans, George Peter Banyard, Ilija Tovilo, Jakub Zelenka, Máté Kocsis, and Arnaud Le Blanc for an interview.

## Sebastian: Let's start with a round of introductions. Who are you?

**George:** Hello, my name is George Peter Banyard. I'm studying pure mathematics at Imperial College London and expected to graduate in October. I'm half German, half British, but grew up in the south of France. During my free time I mostly watch anime or play puzzle games and Richii mahjong.

**Derick:** I am originally from the Netherlands but currently living and enjoying London in the United Kingdom. When I am not behind my computer I will be likely be walking in the countryside or site, with my trusty camera to commit these views to screen.

**Ilija:** I am from Bern, Switzerland. I grew up here, but also have roots and relatives in Croatia. When I am not working, I enjoy spending time with family and friends, being in the sun, and listening to music. I also enjoy solving puzzles and playing chess.

**Jakub:** I am originally from Prague. I moved to the UK over 10 years ago. I work as a contractor and spend most of my free time with my family. I have got 2 dogs that I often walk.

**Máté:** I am a 31-year-old software developer, based in Hungary. Currently, I work for LastPass, and I am responsible for keeping our PHP backend modern as well as bringing it into the cloud. In my free time, I love riding my bike, walking, travelling. But my greatest hobby is certainly programming.

**Arnaud:** I live in France and work at a SaaS company doing awesome things with PHP. When I'm not working there or on PHP, I tend to be in front of my computer or on a bicycle.

# PHPはどこに行くのか

- おそらく、どの言語よりも利用者の幅が広い言語
  - 動的なホームページを作りたい個人
  - Web制作からのステップアップ
  - CMSを使ったWebサイト構築
  - 各種システム開発（大規模Webサイト、ゲームのAPIサーバ…）



# おそらく溝は広まりつつある

- PHPの先進的な言語機能を取り入れて使っているか
- PHPの品質保障(テスト・静的解析)のトレンドに乗れているか
- PHP 8.xへのアップデートはかなり大変
  - PHP 7以前よりもエラーがかなり厳格化した
- レンタルサーバーやVPSを放置している人もおそらく多い



## PHPから後方互換性を除いた「P++」が提案される

ストーリー by hylom 2019年08月16日 17時30分 フォークしたほうが良いんじゃない、と思ったがそういうものはすでにあった 部門より

Anonymous Coward曰く、

型宣言の導入など近年大きな変更の続くPHPだが、コミュニティ内での互換性を巡る議論の中で、後方互換性に縛られない革新的な言語「P++」を作る提案が出されたという ([まとめ記事](#)、[FAQ和訳](#))。

この提案は、開発者の[Zeev氏](#)が「[Bringing Peace to the Galaxy \(銀河に平和をもたらす\)](#)」というタイトルでメーリングリストに投稿したもの。PHPでは7以降たびたび大きな変更が入るとともにレガシーな機能の廃止も行われているが、こうした互換性をなくす変更に対しては、コミュニティ内に反発もあるのだという。そこで氏は、互換性を保ったPHPを開発する一方、革新的な機能を取り込むための言語 (仮称:P++) を提案したとのこと。P++はPHPのforkではなく、あくまでPHPの別モードとなる想定のようなのだ。

ただし、この提案は現状コミュニティからは「開発リソースが無い」という理由などから否定的な反応が寄せられているとのことで、近い将来にP++が誕生することは無さそうである。

49 コメント



php

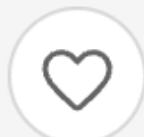


プログラミング



programming





515



200



📌 この記事は最終更新日から3年以上が経過しています。



@tadsan (園島 ぬぬ)

## P++: 銀河に平和をもたらすための奇策と決着

PHP

p++

最終更新日 2019年08月15日 投稿日 2019年08月14日

PHP 8から、PHPは「PHP」と「P++」という2つの言語を提供するようになる

というキャッチーな紹介をするP++: 静的型付けをめざすPHPという記事がそれなりに話題になり、このニュースは目覚しく革新的な内容で、多くのひとの目を引き付けました。

これは**早まった理解**であり、ほとんど誤報と言ってもいい内容でした。2019年8月15日には提案者本人も、少なくとも「P++」の計画を短期的に実現するとは非現実的であり時期尚早であることを認めています。

この記事では、PHP開発の現状、なぜ野心的なP++計画が提案され、事実上撤回されたかの経緯について紹介します。



internals@(メーリングリスト)の最近の(それとあまり最近ではない)議論と最近の提案のいくつか、そして言語の方向性についてのいくつかの声明の見てください。明らかに私たちは両極化が高まっています。

Peterが昨日言ったように、一部のひとたちはいくつかのレガシーなものを取り除きたいのです。私が思うに実際にはその範疇を超えてしまっており、internals@の多くは、PHPの各部はひどく修理が必要で(私も一部同意しますが)、ほかの競合する言語にある機能が「彼らの意見では」非常に欠けているとのことです。

両極化した一方の端には、私たちが常に抱いていた、後方互換性についての強い偏見を維持すべきであり、PHPは緊急のオーバーホールを要しているわけではなく、機能に関して「少ないことは豊かなこと (less is more)、私たちはほかの言語から競って機能をコピーすることはせず、むしろPHPをシンプルに保つことを選択する」と考える人々が居ます。

これらの視点はほとんど正反対です。そのため、internals@の議論は片方が勝って相手が負ける**ゼロサムゲーム**と化してしまいました。

# PHPは過去と未来を繋ぐ言語

- PHPの仕様はCGIの時代から地続き
  - PerlやRubyはApache mod時代やPost WSGI時代に断絶がある
  - PHPはpreforkなサーバ上でもCGI互換のシェアードナッシング  
(すべての状態が隔離され、リクエストごとに初期化される)
- 汎用プログラミング言語の能力を持ちながら根幹がテンプレートエンジンにあるので、小規模からある程度大規模なアプリケーションにもフィットする

# Webの世界のトレンド

- シェアを伸張する各言語
  - Node.js, SPA
  - パフォーマンスに優れた静的型付き言語
- 相対的にPHPの存在感は薄れつつある
  - このままPHPは滅びるしかないのか…？



# PHPはどこでも65点がとれる言語

- 「現代におけるプロダクト開発とPHPを選定するワケ #phpkansai」
  - By @potato4d (PHPカンファレンス関西2017)
- 小難しいことを考えずに簡単にWebアプリを書ける(HTML+α)の言語
  - なんだかんだいってWEB+DB連携がさくっとできる言語ランタイムとしての存在感は健在
- 求められているのは素晴らしい言語ではなく簡単に使える言語

PHP先生の  
雑草のような生命力に  
ご期待ください！