ぺちこん北海道
学生ランチセッション

PHP for Students

PHPカンファレンス北海道 実行委員会
USAMI Kenta

2024-01-13
PHPカンファレンス北海道2024

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）

tadsan loves 北海道

- 北海道砂川市出身
- 北海道工業大学卒業（現・北海道科学大学）
- 2011年1月～2012年7月くらいまで札幌のコミュニティに顔を出してた
  - LOCAL学生部
  - Ruby札幌
  - Python札幌

tadsanとPHP

- 北海道に住んでたときはほとんどPHPを書いていない(!)
- 2010年にセキュリティキャンプに参加してからプログラミング言語に興味
- PHPは2012年に現職に入社してからやりはじめた
- PHPカンファレンス北海道2016が初めてのカンファレンス発表
  - Webフレームワークを作る話とか静的解析とか、脆弱性の話をしています

<?php

さて
PHPは何ができる
言語なのでしょうか

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

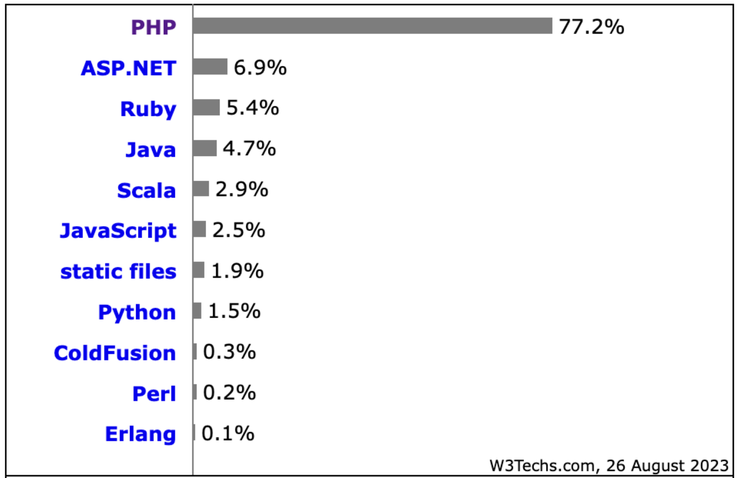
Usage statistics of server-side programming languages for websites

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.

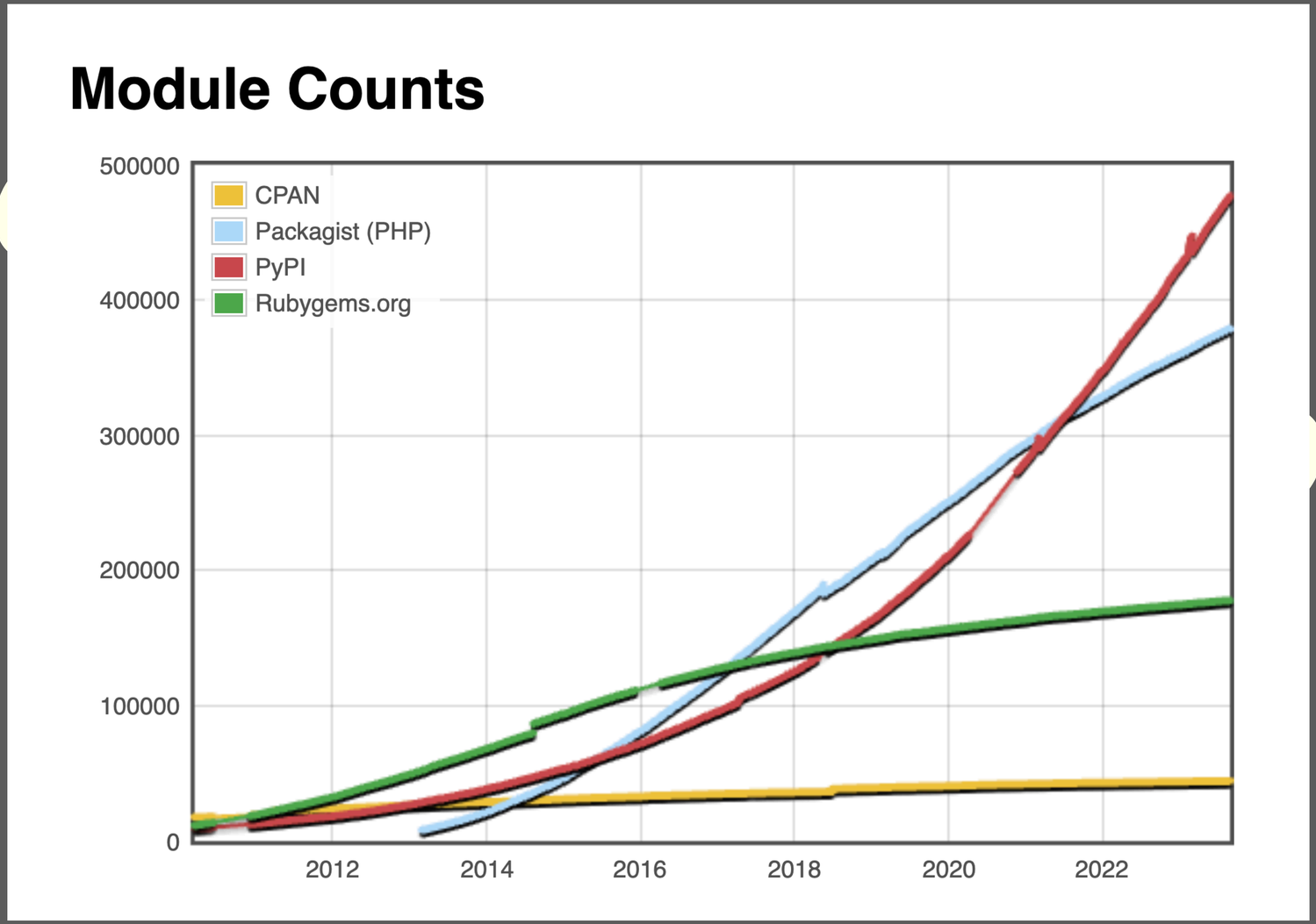
How to read the diagram:
PHP is used by 77.2% of all the websites whose server-side programming language we know.



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
500000
400000
300000
200000
100000
0
CPAN
Packagist (PHP)
PyPI
Rubygems.org
2012
2014
2016
2018
2020
2022

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（得意とは言ってない）
  - CLI
  - GUIもいちおう…（得意とは言ってない）

みなさんの想像する PHPとは

A

```
<!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>
```

B

```php
<?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>';
```

C

```php
<?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

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

整理しましょう

A=SSIスタイル

```
<!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>
```

Perl

```
use DBI;

$db = DBI->connect('...');
$stmt = $db->prepare('SELECT * FROM books');
$stmt->execute;
print '<h1>本の一覧</h1>'; print '<ul>';
for ($i = 0; $I < $sth->rows; $i++) {
    @row = $sth->fetchrow_array;
    print '<li>'; print $row[0]; print '</li>';
}
print '</ul>';
```

B=CGIスタイル

```
<?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>';
```

C=OOPスタイル

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

namespace Foo\Http\Controller;

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

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

PHPは汎用プログラミング言語

- PHPはHTTP(≒Webサービス)を言語レベルでサポートしている
  - ほかの言語ではCGIライブラリとしてサポートしている機能が組み込み
- 各Webサーバとの連携は**SAPI**(Server API)として定義されている
  - どのサーバー上でもCGI風の動作をする
  - コマンドラインスクリプトとして実行するモード(CLI)もSAPIのひとつ

PHPの歴史

- 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

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

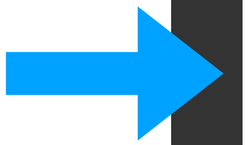
- PHP 7.4: 型付きプロパティ
- PHP 8.0: コンストラクタプロモーション（PHP 8.2）
- PHP 8.1: readonlyプロパティ
- PHP 8.2: trait定数宣言
- PHP 8.3: 型付き定数宣言

```php
<?php

class Book {
  /**
   * @var string
   * @readonly
   */
  private $name;

  /** @param string $name */
  public function __construct($name) {
    $this->name = $name;

  ...
```



```php
<?php

class Book {
    public function __construct(
        public readonly string $name,
    ) {
    }

 ...
```

静的解析しやすい特徴

- クラスや関数が再定義されることは基本的にない
- Java風のクラス継承（class, interface）
- 関数呼び出しは全て（）がつく、文末は必ず；

PHPの進化は
型宣言の進化

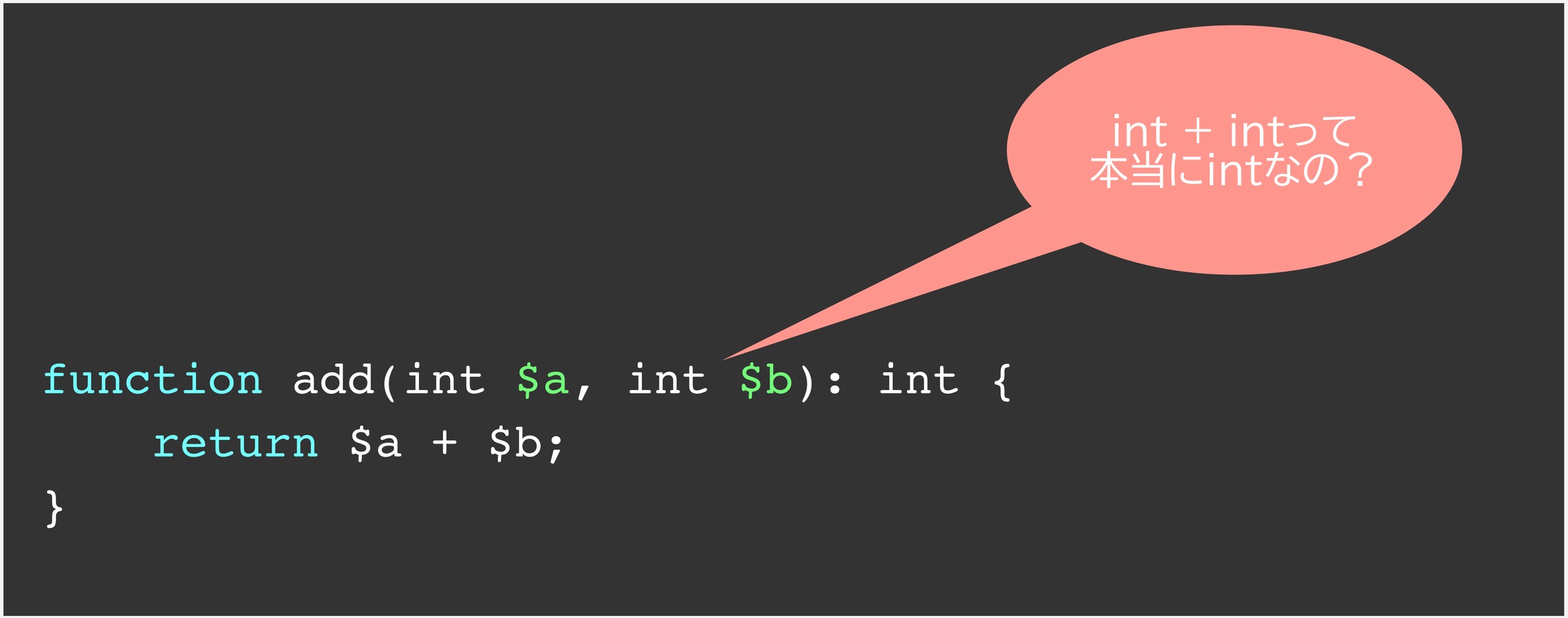
いまやPHPは静的型付きと言っても過言ではない

(本当か…？)

型がついていない関数(PHP5)

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

スカラー型宣言(PHP7)




```php
function add(int $a, int $b): int {
    return $a + $b;
}
```

広い値をとるにはfloatが必要

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

ひとつの解決策では
あるが… 不必要にfloat
を強制するのか

PHPDocの型注釈(アノテーション)

```php
/**
 * @param int|float $a
 * @param int|float $b
 * @return int|float
 */
function add($a, $b) {
    return $a + $b;
}
```

ユニオン型宣言 (PHP8.0)

```php
function add(int|float $a, int|float $b): int|float {
    return $a + $b;
}
```

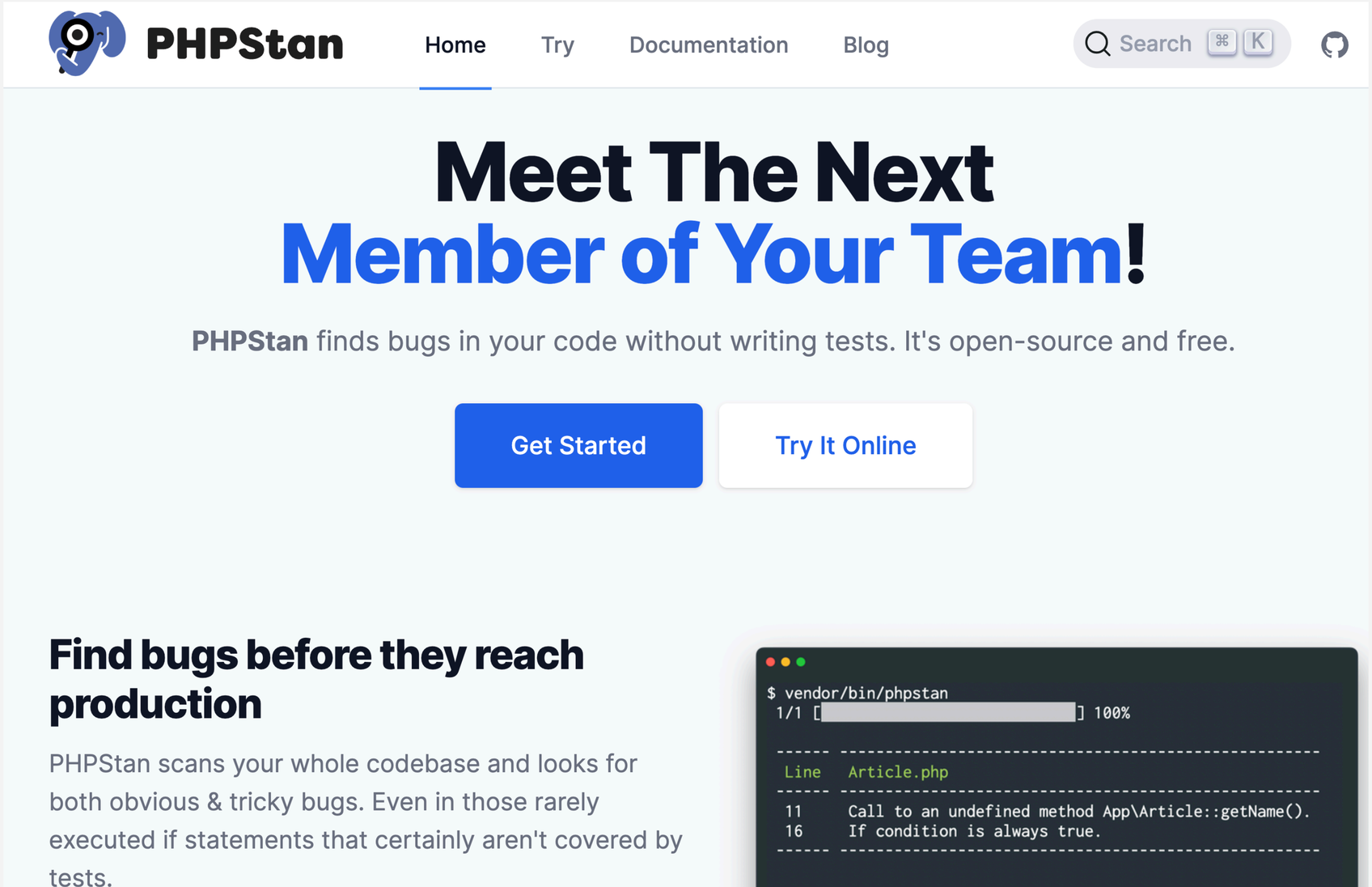
LL言語としてのPHP vs エディタ

- 2000年代のCGIやLL言語（Lightweight Language）という用語がよく使われていた頃は「重厚鈍重なIDEが必要な言語」のアンチテーゼとしてシンプルなスクリプト言語とエディタが好まれていた（ように思う）
- 時は流れ、Webアプリケーションは巨大になり、ファイルは増え、シンプルなエディタでは開発のオーバーヘッドが無視しがたくなってきた
- PhpStorm（JetBrains社のIDE）が静的解析とインテリジェントな編集機能を提供してくれるようになった

PHPと静的解析

- PHP組み込みで提供されているのは php -l（syntax check）のみ
- 「静的解析ツール」と呼ばれうるものは2000年代からあった
  - ドキュメント生成、コーディングスタイルのチェック
- 2010年代後半になってPhpStormの普及を呼び水にして、
  型システムや変数スコープを正確に扱えるツールが開発され始めた

PHP Static Analysis Tool



PHPStanとは

- 2016年から開発されているPHPの静的解析ツール
  - Ondřej Mirtesさんの個人プロジェクト、2021年からフルタイム開発
- 開発当初は純粋な静的解析ではなく実行時リフレクションを用いることで高速な解析を実現していた
  - 現在は静的解析がデフォルトで、レガシープロジェクトに導入しやすくなった
- その他のPHP静的解析ツールにはPsalm, Phan, Qodana（PhpStorm）

PHPStanの提供する型

- PHPの組み込み型、クラス名、インターフェイス名
- 修飾型: non-empty-string, non-empty-array
- ユニオン型: A|B
- ジェネリクス: array<T>, ArrayObject<T>
- 定数型: 123, ”foo”, Foo::BAR

PHP処理系

- ソースコード: https://github.com/php/php-src
- 処理系のソースコードはC言語で書かれている（一部のみC++）
  - パーサーはYacc（Bison）で記述されている
  - 基礎知識は『Rubyソースコード完全解説』の知識が役に立つ
- 議論はメーリングリストがメインだが、GitHub issueやPull Requestも受け付けてくれる

PHPの開発体制

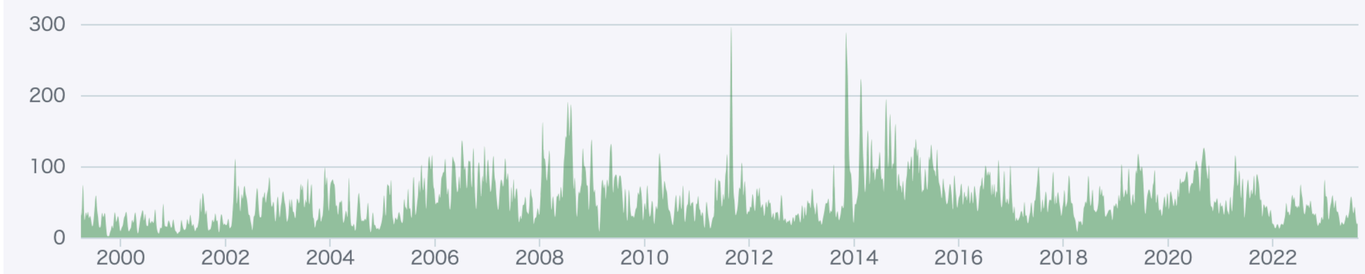
- Request for Comments https://wiki.php.net/rfc
  - 言語機能の改廃提案と投票プロセス、リリース周期が明文化
- 機能提案する人は仕様をまとめてMLで議論、実装も責任を持って用意する
- PHP原作者のRasmus Lardorfは開発者としての1票しか行使しない
- PHP 7.0（2015年）以降、11月末〜12月初頭にリリースが定着
  - 品質保障のためのfeature freeze、RC版なども定められた

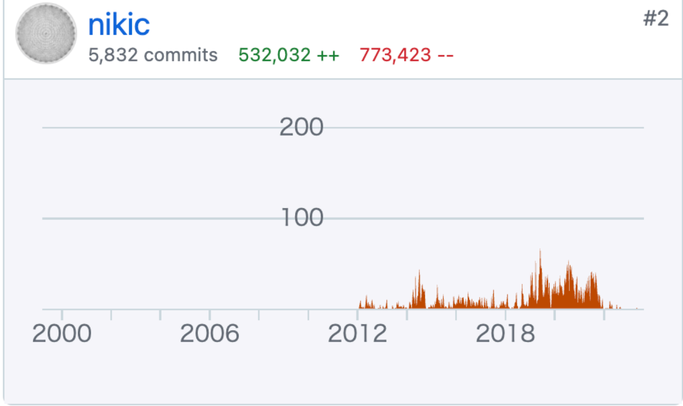
PHPは誰が開発しているのか

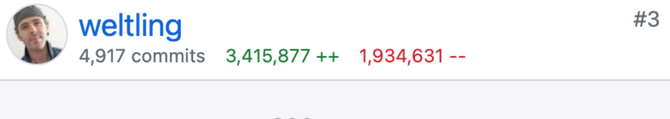
Apr 4, 1999 – Aug 26, 2023

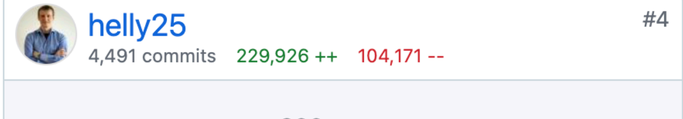
Contributions: Commits

Contributions to master, excluding merge commits and bot accounts









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

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

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

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…）

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！