後ろ向きにならないPHP7から8へのアップグレード

2023年12月6日

一般社団法人BOSS-CON JAPAN
PHP技術者認定機構
エバンジェリスト 三雲 勇二

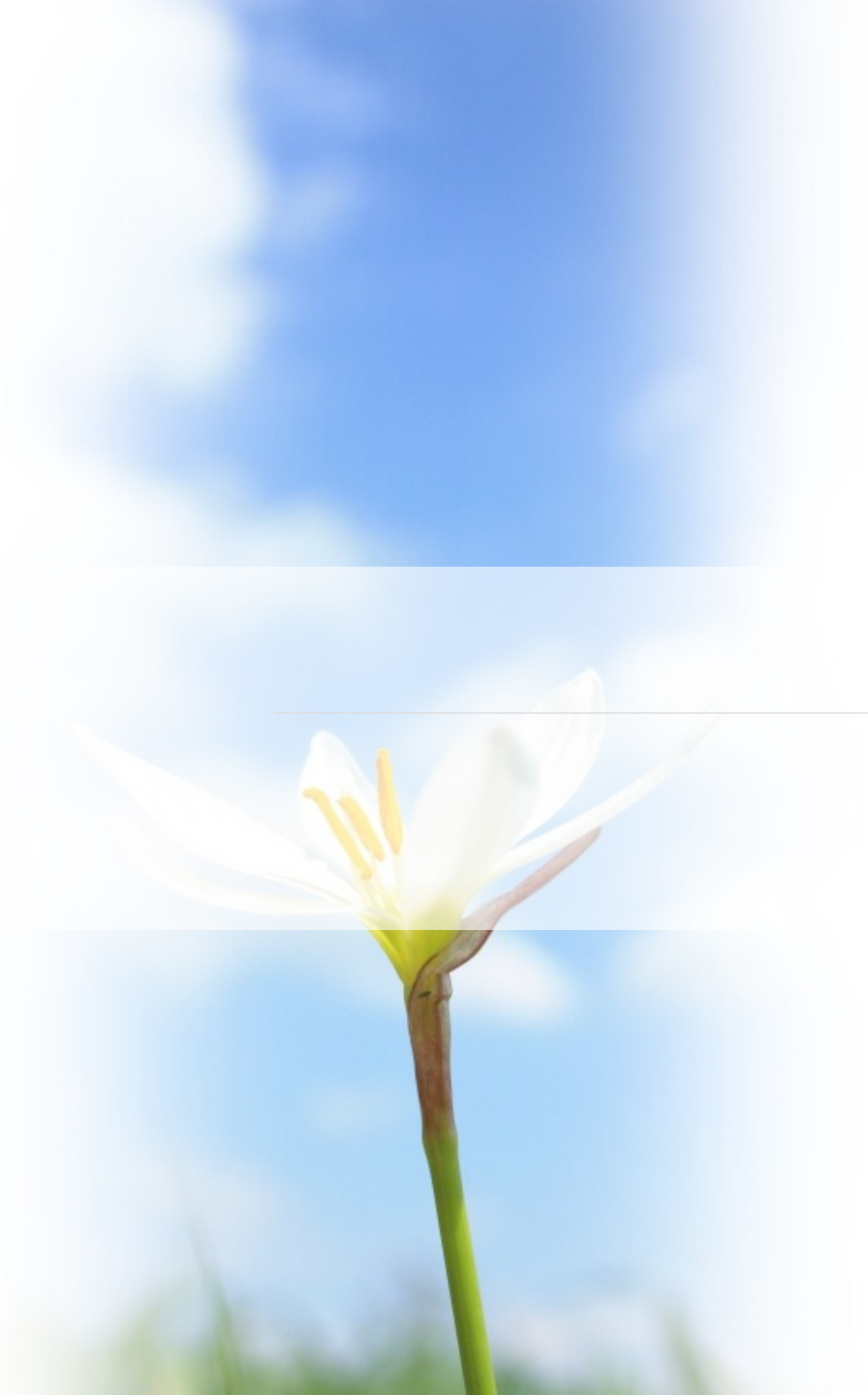
Copyright © PHP技術者認定機構 All Rights Reserved.

アジェンダ

- 自己紹介
- 【本編】
  後ろ向きにならないPHP7から8へのアップグレード

  1. 改めて確認：PHP 7 から 8 の変更点
  2. ツールを使って継続的にアップグレード
- まとめ

Copyright © PHP技術者認定機構 All Rights Reserved.



自己紹介

Copyright © PHP技術者認定機構 All Rights Reserved.

自己紹介

- 三雲 勇二 （みくも ゆうじ）

  三雲 勇二 / yuji.mikumo

  mikumo [@maikeru]

- 所属：
  - プライム・ストラテジー株式会社
    WordPress などCMS高速化技術
    KUSANAGI の会社です😊
  - PHP技術者認定協会 エバンジェリスト

- PHP 8 上級 / PHP 5 上級 / 徳丸実践 / KUSANAGI for WordPress ほか

超高速CMS実行環境
KUSANAGI
Powered by Prime Strategy

Copyright © PHP技術者認定機構 All Rights Reserved.

後ろ向きにならない
PHP 7 から 8 へのアップグレード

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP 8 で抑えておくポイント

1.改めて確認：PHP 7 から 8 の変更点

2.ツールを使って継続的にアップグレード

Copyright © PHP技術者認定機構 All Rights Reserved.

改めて確認：PHP 7 から 8 の変更点
初級者に抑えておいてほしいポイント

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP 8 で抑えておくポイント

1. エラーレベルの変更
2. 型プログラミング
3. 新機能や新しい構文

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP 8 で抑えておくポイント

1. エラーレベルの変更
2. 型プログラミング
3. 新機能や新しい構文

PHP 7 系以下は、すべて EOL（サポート切れ）になっていますので、特に PHP 8 系で動作させる部分を中心に説明いたします。

Copyright © PHP技術者認定機構 All Rights Reserved.

エラーレベルの変更

【よくある質問】

PHP 8 系にバージョンアップしたところ

Fatal Error が表示される

Copyright © PHP技術者認定機構 All Rights Reserved.

エラーレベルの変更

エラーのレベルが上っているものが多いため。

- 警告 Warning → エラー Error
- 注意 Notice → 警告 Warning

Copyright © PHP技術者認定機構 All Rights Reserved.

エラーレベルの変更

- Notice: 注意。軽微な問題です。ウェブサイトによっては対応しなくても良いと書かれていることもありますが、基本は対応することが望ましいです。
- Warning: 警告。PHP の実行は継続しますがエラーです。
- Fatal Error: エラー。PHP の実行が中断されるエラーです。
- Parse Error: エラー。PHP の文法エラーです。PHP の実行はできませんので中断されます。
- Deprecated: 非推奨。PHP の実行は継続されます。非推奨の関数などを使用している際に表示されるエラーです。将来的な PHP でエラーとなる可能性があるので対応が必要です。
- Strict Standards: 古い構文など。PHP の実行は継続されます。PHP の古い構文を使用している場合に表示されます。将来的な PHP でエラーとなる可能性があるので対応が必要です。

Copyright © PHP技術者認定機構 All Rights Reserved.

エラーレベルの変更

エラー表示のデフォルト値が異なります。

（php.ini の error_reporting の初期値）

- PHP8系: `E_ALL` 、すべてのエラー表示。
- PHP7系: `E_ALL & ~E_NOTICE & ~E_STRICT & ~E_DEPRECATED` 、Notice と Strict、Deprecated のエラーは表示しないため、すべてのエラーを表示するわけではない。

Copyright © PHP技術者認定機構 All Rights Reserved.

エラーレベルの変更

PHP8に移行する前に確認すべきこと

PHP7系以前で、php.ini 設定等で error_reporting エラーレベルを「すべて」に変更し、エラーログを確認、できる限りエラーを消しておく。

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

【よくある質問】

文字の「‘1’」と、数字の「1」、

同じ扱いではダメですか？

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

内部では別物です。

16進数バイトという表記で表すと、

- 数値 1 →「1」
- 文字 1 →「31」(ASCII)

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

内部では別物です。

16進数バイトという表記で表すと、

- 数値 1 →「1」
- 文字 1 →「31」(ASCII)

「1」と「31」という内部のデータが違うものを、PHPが気を利かせて同じ「1」として取り扱っていてくれていたのです。

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

```
// 型の自動変更を行ったうえで比較されます
1 == '1'; // true

// そのまま比較されます
1 === '1'; // false
```


Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

```
// 型を意識していないと、意図した動作なのかわかりません。
1 == '1'; // true

// 型を意識すると、意図がわかりやすくなります。
1 === '1';    // false
1 === (int)'1'; // true
```


Copyright © PHP技術者認定機構 All Rights Reserved.

下位互換性のない変更点 – 文字列と数値の比較

(厳密でないやり方で)数値と非数値文字列を比較する場合、 数値を文字列にキャストし、文字列と比較するようになりました。 数値と数値形式の文字列の比較は、以前と同じ振る舞いをします。

緩やかな比較 (==)	変更前 (PHP 7.4 以前)	変更後 (PHP 8.0 以降)
`0 == "0"`	true	true
`0 == "0.0"`	true	true
`0 == "foo"`	true	false
`0 == ""`	true	false
`0 == "not-a-number"`	true	false
`42 == " 42"`	true	true
`42 == "42foo"`	true	false

Copyright © PHP技術者認定機構 All Rights Reserved.

下位互換性のない変更点 – 文字列と数値の比較

(厳密でないやり方で)数値と非数値文字列を比較する場合、 数値を文字列にキャストし、文字列と比較するようになりました。 数値と数値形式の文字列の比較は、以前と同じ振る舞いをします。

厳密な比較 (===)	変更前 (PHP 7.4 以前)	変更後 (PHP 8.0 以降)
`0 === "0"`	false	false
`0 === "0.0"`	false	false
`0 === "foo"`	false	false
`0 === ""`	false	false
`0 === "not-a-number"`	false	false
`42 === " 42"`	false	false
`42 === "42foo"`	false	false

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

下位互換性のない変更点 – 文字列と数値の比較

対策：できる限り、キャスト演算子で比較する型を合わせ、
緩やかな比較（==）ではなく、 厳密な比較（===）を使いましょう。

厳密な比較 （===）	変更前 （PHP 7.4 以前）	変更後 （PHP 8.0 以降）
`0 === "0"`	false	false
`0 === "0.0"`	false	false
`0 === "foo"`	false	false
`0 === ""`	false	false
`0 === "not-a-number"`	false	false
`42 === " 42"`	false	false
`42 === "42foo"`	false	false

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

厳密な比較のメリット [===、!==]

- 高速
- （文字列型以外の場合）インジェクション系のセキュリティ対策

  SQLインジェクション例、`3 === '3 OR 1=1'` のような攻撃を防ぐ。
- Mixed な戻り値を持つ関数の動作を正しく処理できる

  例: strpos の戻り値 `0`(先頭) と `false`(文字列なし) が同一視されない。

緩やかな比較メリット [==、!=、<>]

- 型を意識する必要がない

  初心者など型の概念がわからない場合、とりあえず動くコードが書けます。
- 同一クラスの比較

  同一クラスの別インスタンスを同等と扱うことができます。
- キャストする手間がない

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

型を意識したプログラミングするために、

```
declare(strict_types=1);
```

PHPの暗黙の型変換を禁止して、型指定に厳密にチェックする設定

Copyright © PHP技術者認定機構 All Rights Reserved.

型プログラミング

また、比較演算子以外にも影響があるものがありますので、それらの考慮も必要です。

- switch 文 → match() 式
- in_array() 関数 → 第3引数を true に指定

[訂正]
動画では strpos() 関数となっていますが、
正しくは in_array() 関数です。

Copyright © PHP技術者認定機構 All Rights Reserved.

【よくある質問】

PHP 8 で変更があるものは？

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある機能

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある機能

- JIT
- 名前付き引数
- アトリビュート
- union 型
- match 式
- nullsafe 演算子
- WeakMap クラス
- ValueError クラス
- トレイトの抽象メソッド
- ::class 構文がオブジェクト
- Stringable インターフェイス
- マイナススタートの配列インデックス
- throw 式

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

- 他に引数に NULL を渡すと挙動が変わる関数例

  strpos, mb_strpos: $needle 引数

  substr: $length 引数

  array_splice: $length 引数

  mktime: 各引数

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

strpos() 関数の例 [https://www.php.net/manual/ja/function.strpos]

strpos

(PHP 4, PHP 5, PHP 7, PHP 8)
strpos — 文字列内の部分文字列が最初に現れる場所を見つける

needle
PHP 8.0.0 より前のバージョンでは、**needle** が文字列でない場合、 数値に変換され、文字の通常の値として扱われていました。 この振る舞いは PHP 7.3.0 以降では推奨されないので、 この機能を使用しないことを強く推奨します。 意図した動作に依存する場合、 **needle** を string に明示的にキャストするか、 明示的に chr() 関数を呼び出すべきでしょう。

変更履歴

バージョン	説明
8.0.0	**needle** に数値を渡すことはサポートされなくなりました。
7.3.0	**needle** に数値を渡すことは非推奨になりました。
7.1.0	負の **offset** をサポートするようになりました。

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

つまり strpos() 関数では、

- $needle 引数には文字列しか渡せなくなるので、事前に chr() 関数などで文字列に変更しておく。
- $needle 引数に NULL と空文字で戻り値に違いあり。ドキュメントにない。
  - PHP8: 0
  - PHP7: false

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

```
※ PHP 8.1 ($needle=NULL)
Deprecated: strpos(): Passing null to parameter #2 ($needle) of type
string is deprecated in /home/user/php/code.php on line 3
int(0)

※ PHP 8.0
int(0)

※ PHP 7.3 / 7.4
Deprecated: strpos(): Non-string needles will be interpreted as
strings in the future. Use an explicit chr() call to preserve the
current behavior in /home/user/php/code.php on line 3
bool(false)

※ PHP 7.2 以前
bool(false)
```


Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

対処が難しい関数や、取り急ぎ対応する場合、

PHP7以前の動きをする関数を用意する。

```
function php7_strpos($haystack, $needle, $offset = 0)
{
    if ($needle === '' || $needle === null) {
        // エラーログにヒントを出力
        $err = '$needle = ' . ($needle === null ? 'NULL' : "''");
        error_log("PHP7: strpos: $err");
        error_log(var_export(debug_backtrace(), true));

        // 以前の動作
        return false;
    }
    return strpos($haystack, $needle, $offset);
}
```


Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

ログに出力しておくと、後の改修時の資料になる。

```
PHP7: strpos: $needle = NULL
array (
  0 =>
  array (
    'file' => '/home/user/php/code.php',
    'line' => 3,
    'function' => 'php7_strpos',
    'args' =>
    array (
      0 => 'test',
      1 => NULL,
      2 => 0,
    ),
  ),
)
```


Copyright © PHP技術者認定機構 All Rights Reserved.

PHP8で変更のある関数

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHP バージョンのアップグレードでツールを使うと、ソースコードのチェックをしてもらえます。

本日は３つご紹介します。

- PHP Compatibility Coding Standard for PHP CodeSniffer
- PHPStan
- Rector

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHP Compatibility Coding Standard for PHP CodeSniffer

https://github.com/PHPCompatibility/PHPCompatibility

PHP Compatibility Coding Standard for PHP CodeSniffer

stable 9.3.5 | unstable dev-develop | awesome? yes! | license LGPL-3.0-or-later | Flattr this git repo

CS passing | Test passing | coverage 98%

php >=5.3 | tested on PHP 5.4 | 5.5 | 5.6 | 7.0 | 7.1 | 7.2 | 7.3 | 7.4 | 8.0 | 8.1 | 8.2 | 8.3 | nightly

This is a set of sniffs for PHP CodeSniffer that checks for PHP cross-version compatibility. It will allow you to analyse your code for compatibility with higher and lower versions of PHP.

- PHP Version Support
- Requirements
- Thank you
- Upgrading to PHPCompatibility 9.0.0
- Installation in a Composer project (method 1)
- Installation via a git check-out to an arbitrary directory (method 2)
- Sniffing your code for compatibility with specific PHP version(s)
  - Using a framework/CMS/polyfill specific ruleset
- Using a custom ruleset
  - `testVersion` in the ruleset versus command-line
  - PHPCompatibility specific options
- Projects extending PHPCompatibility
- Contributing
- License

PHP Version Support

The project aims to cover all PHP compatibility changes introduced since PHP 5.0 up to the latest PHP release. This is an ongoing process and coverage is not yet 100% (if, indeed, it ever could be). Progress is tracked on our GitHub issue tracker.

Pull requests that check for compatibility issues in PHP 4 code - in particular between PHP 4 and PHP 5.0 - are very welcome as there are still situations where people need help upgrading legacy systems. However, coverage for changes introduced before PHP 5.1 will remain patchy as sniffs for this are not actively being developed at this time.

Packages

No packages published

Used by 37.1k

+ 37,109

Contributors 48

+ 34 contributors

Languages

PHP 100.0%

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHP Compatibility Coding Standard for PHP CodeSniffer

https://github.com/PHPCompatibility/PHPCompatibility

PHP でチェックツールといえば phpcs と言われるほど有名なツールがあります。

この phpcs に追加する形で、PHP のアップグレードのチェックができるようになります。

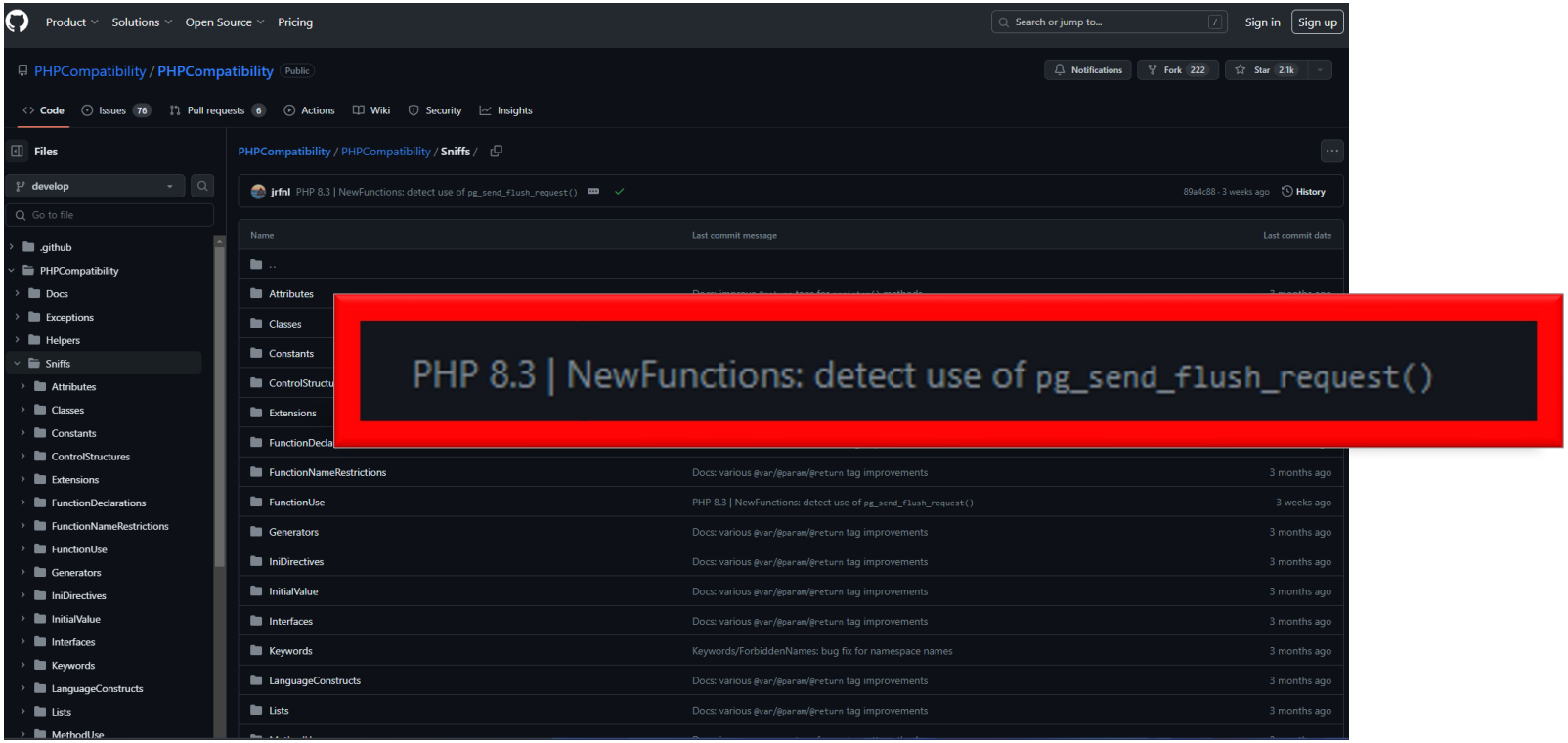
また、phpcbf で自動変換もできるようになります。

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHP Compatibility Coding Standard for PHP CodeSniffer

https://github.com/PHPCompatibility/PHPCompatibility

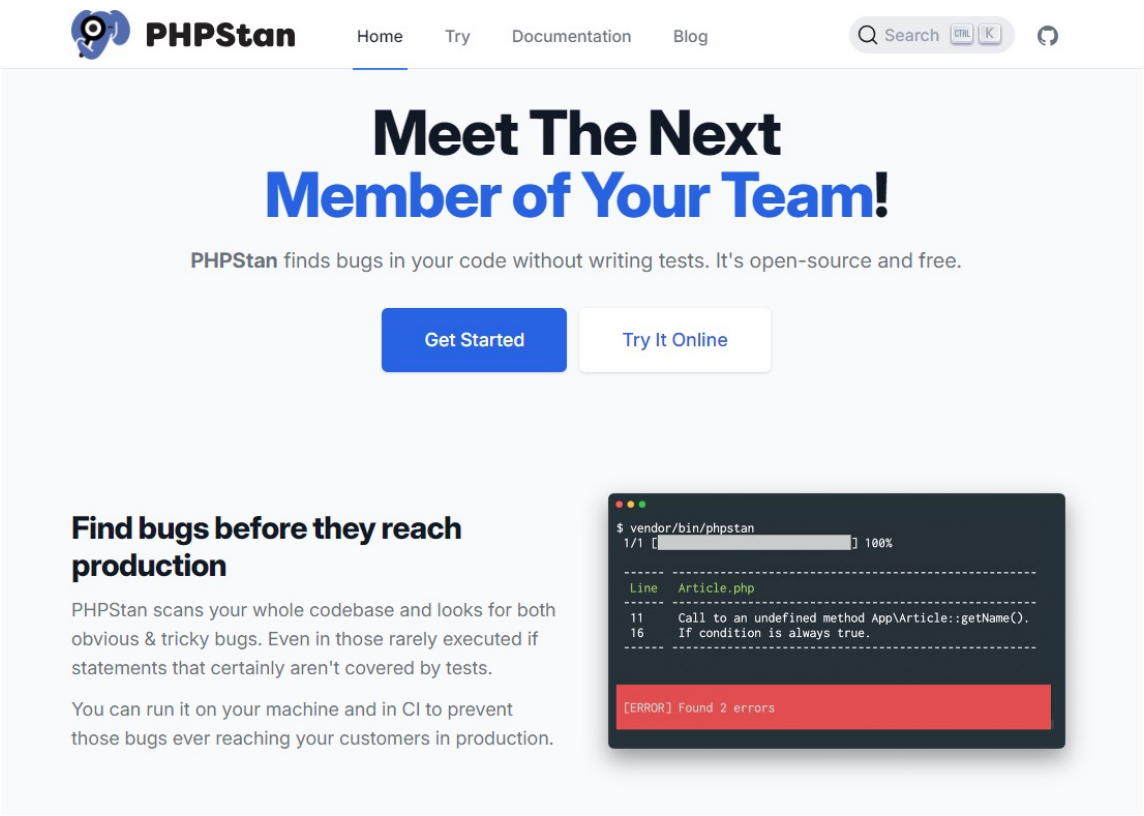


Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHPStan

https://phpstan.org/



Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHPStan

https://phpstan.org/

PHP ツールのもはやデファクトスタンダードと言ってもいいツールです。

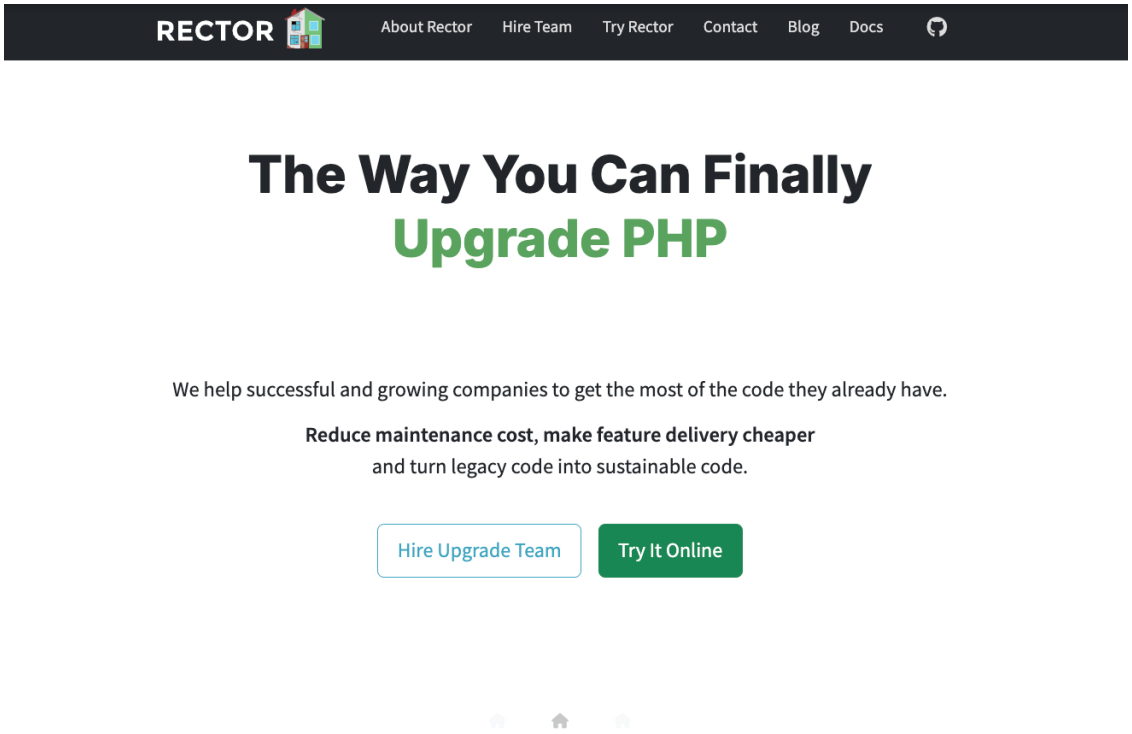
Web 上に情報も多いため、まず使用してみるツールとしてはこちらを検討する価値があります。

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

Rector

https://getrector.com/



Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

Rector

https://getrector.com/

あまり Web 上には情報がないですが、バージョンアップ作業のときに非常に使えるツールです。

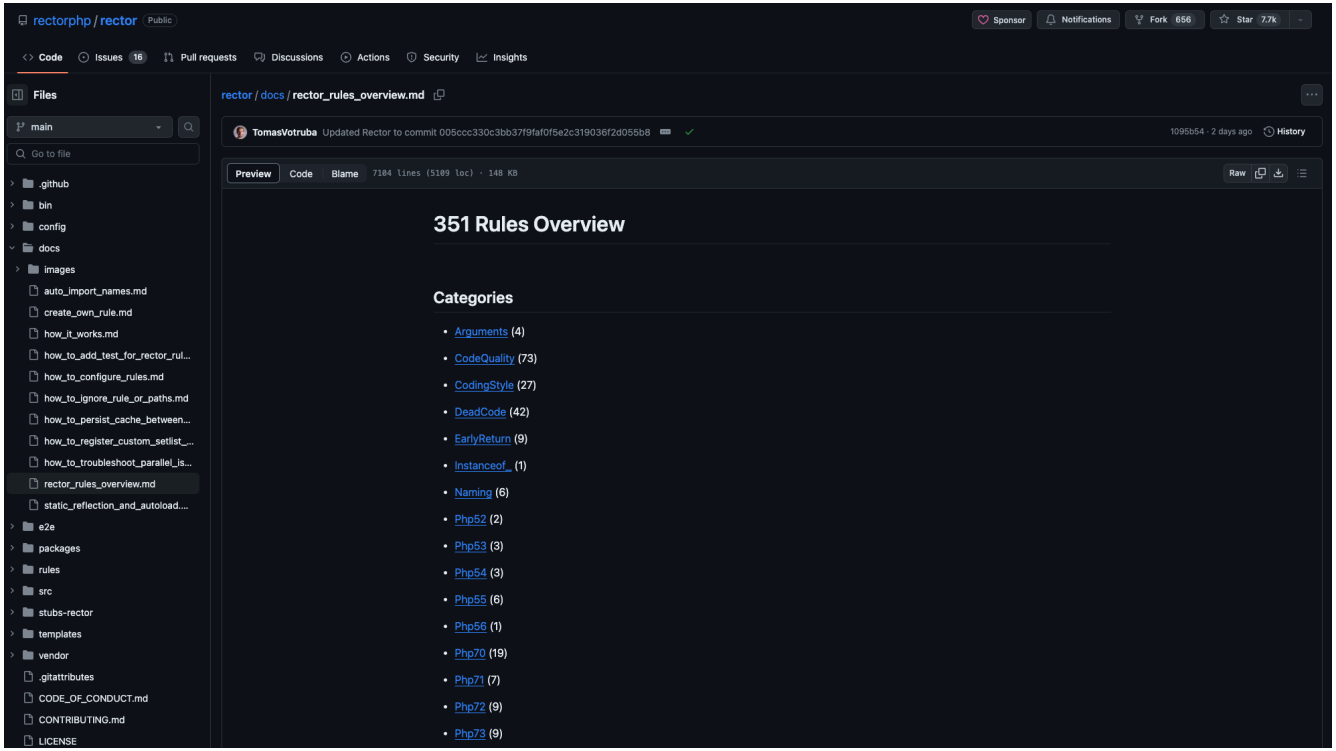
PHPStan とも連携ができるようですので、補完的に使ってみると良いと思います。

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

Rector - ルール

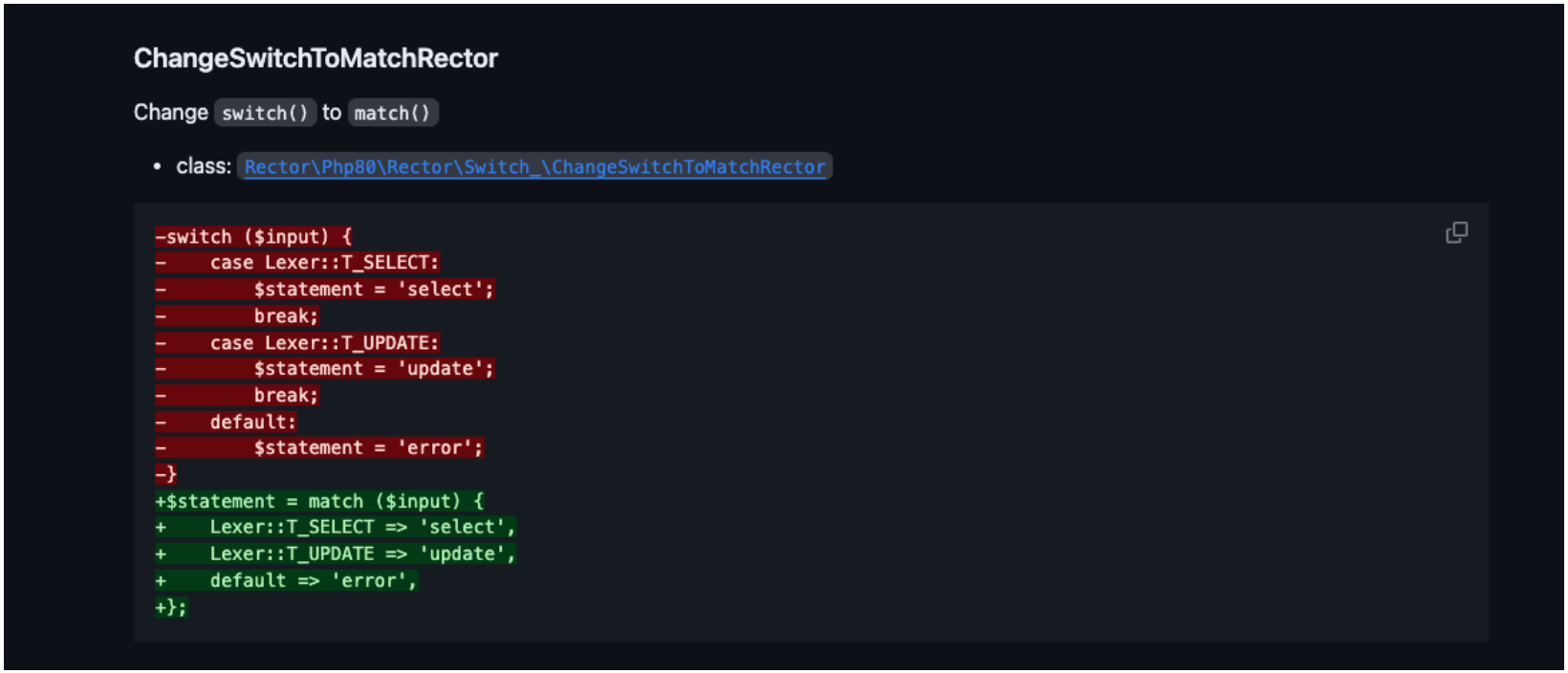
https://github.com/rectorphp/rector/blob/main/docs/rector_rules_overview.md



Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

switch から match に移行するルール



Copyright © PHP技術者認定機構 All Rights Reserved.

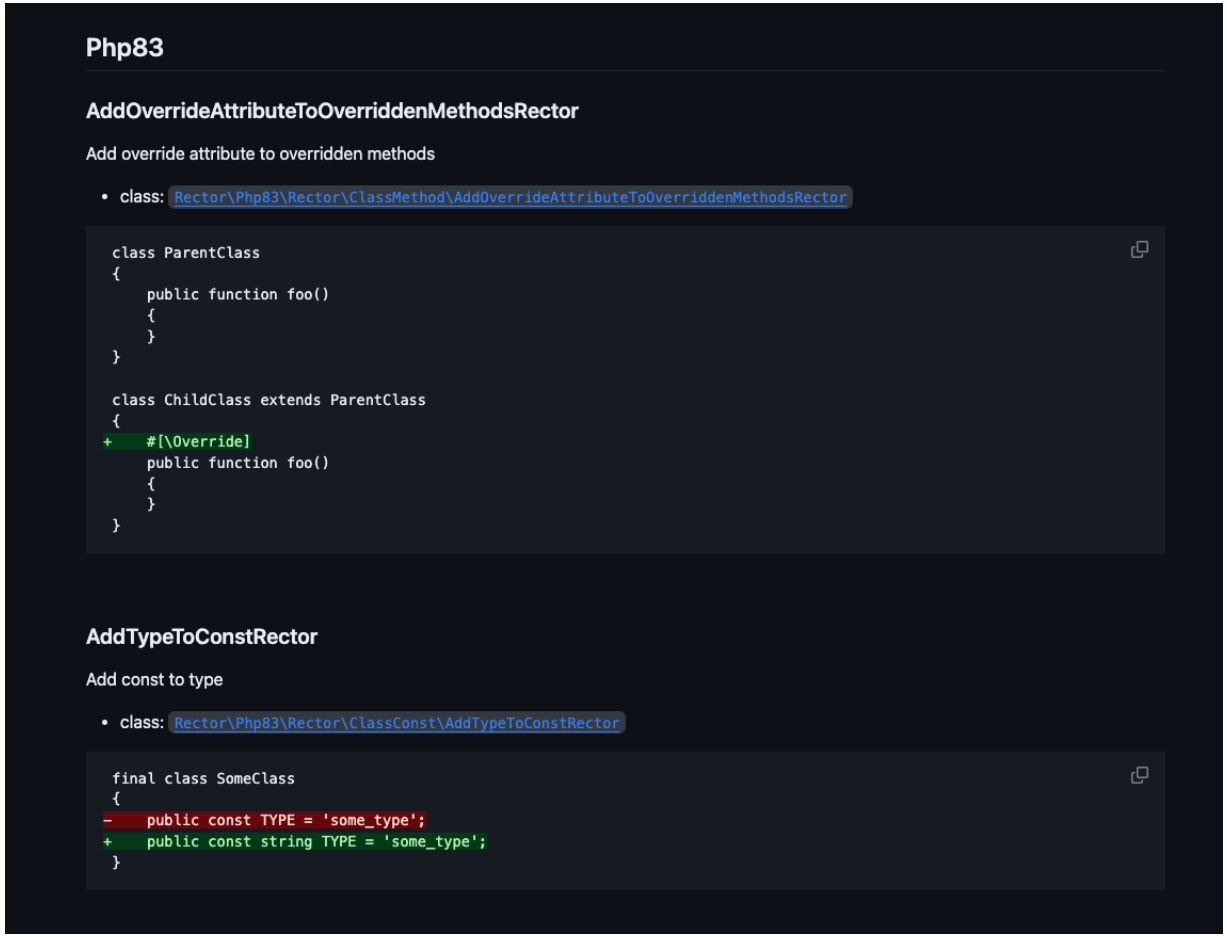
ツールを使って継続的にアップグレード

PHP4 のときのコンストラクタを書き換えるルール

Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

PHP 8.3 ルールも入っています



Php83

AddOverrideAttributeToOverriddenMethodsRector

Add override attribute to overridden methods

- class: `Rector\Php83\Rector\ClassMethod\AddOverrideAttributeToOverriddenMethodsRector`

```
class ParentClass
{
    public function foo()
    {
    }
}

class ChildClass extends ParentClass
{
+    #[\Override]
    public function foo()
    {
    }
}
```

AddTypeToConstRector

Add const to type

- class: `Rector\Php83\Rector\ClassConst\AddTypeToConstRector`

```
final class SomeClass
{
-    public const TYPE = 'some_type';
+    public const string TYPE = 'some_type';
}
```


Copyright © PHP技術者認定機構 All Rights Reserved.

ツールを使って継続的にアップグレード

ツールのまとめ

- **PHP Compatibility Coding Standard for PHP CodeSniffer**

  phpcs 使われている方におすすめ。知見も多いです。
- **PHPStan**

  どのツールを使うか迷っている方にまずおすすめです。
- **Rector**

  情報が少ないですが、細やかな制御ができます。

Copyright © PHP技術者認定機構 All Rights Reserved.

PHP技術者認定機構