

Better PHP

Keep your code up to date



Who Am I?

- Werner „wmk“ Krauß
- Freelancer
- netwerkstatt, Bad Ischl, AT
- PHP since 1998 (PHP3)
- Silverstripe CMS since 2009 (SS 2.3)

The „Problem“

- New PHP version every year
- Language evolves, becomes even more mature
- Cool and useful features are added
- We're still stuck with older PHP versions
 - Modules should also run with older installations
 - No knowledge of new features

What's New In PHP

since 5.6



Let's Dive Into It

Look at a practical example



What Type To You Prefer?

- Return types (7.0)
- Typed parameters
- Object types, nullable types (7.1)
- Typed properties (7.4)
- ...
- Union types
- Intersection types

Sugar For The Baby: New, Smooth Operators

- Null coalesce ?? (7.0)
- Null coalesce assignment operator ??= (7.4)

```
class SomeClass
{
    public function run($values, $keyToMatch)
    {
-       $result = array_key_exists($keyToMatch, $values) ? $values[$keyToMatch] : null;
+       $result = $values[$keyToMatch] ?? null;
    }
}
```

```
$array = [];
-$array['user_id'] = $array['user_id'] ?? 'value';
+$array['user_id'] ??= 'value';
```

Sort Of Star Trek

- Spaceship operator <=> (7.0)
- Can even sort more params at once

```
function order_func($a, $b) {  
-   return ($a < $b) ? -1 : (($a > $b) ? 1 : 0);  
+   return $a <=> $b;  
}
```

```
$people[0] = ['first' => 'Adam', 'last' => 'West'];  
$people[1] = ['first' => 'Alec', 'last' => 'Baldwin'];  
$people[2] = ['first' => 'Adam', 'last' => 'Baldwin'];  
  
function sorter(array $a, array $b) {  
    return [$a['last'], $a['first']] <=> [$b['last'], $b['first']];  
}  
  
usort($people, 'sorter');
```


New, Safe Operators

- Nullsafe operator ?-> (8.0)

```
// As of PHP 8.0.0, this line:  
$result = $repository?->getUser(5)?->name;  
  
// Is equivalent to the following code block:  
if (is_null($repository)) {  
    $result = null;  
} else {  
    $user = $repository->getUser(5);  
    if (is_null($user)) {  
        $result = null;  
    } else {  
        $result = $user->name;  
    }  
}
```

New Strings (Not Only For My Guitar)

- `str_contains()`
- `str_starts_with()`
- `str_ends_with()`
- More efficient and easier to use than `strpos()`

```
public function run()
{
-   return strpos('abc', 'a') !== false;
+   return str_contains('abc', 'a');
}
```

A close-up photograph of two seashells. The shell on the left is white with a small red heart drawn on its surface. The shell on the right is also white with a red star-like drawing. Both shells are tied together at the top with a blue and red string. In the foreground, a light-colored rock has a black star-like drawing on it. The background is dark with some blurred lights.

Romantic PHP

It's A Match!

- Match
 - Replaces switch or if-else chains
 - Typesafe
 - Kind of lookup table

```
-switch ($input) {  
-  case Lexer::T_SELECT:  
-    $statement = 'select';  
-    break;  
-  case Lexer::T_UPDATE:  
-    $statement = 'update';  
-    break;  
-  default:  
-    $statement = 'error';  
-}  
+$statement = match ($input) {  
+  Lexer::T_SELECT => 'select',  
+  Lexer::T_UPDATE => 'update',  
+  default => 'error',  
+};
```

Arrow Functions

- More functional coding style
- Somewhat similar to JavaScript

```
public function run($meetups)
{
-   return array_filter($meetups, function (Meetup $meetup) {
-       return is_object($meetup);
-   });
+   return array_filter($meetups, fn(Meetup $meetup) => is_object($meetup));
}
```

Built In Interfaces

- Stringable
- Iterable

```
-class SomeClass
+class SomeClass implements Stringable
{
-   public function __toString()
+   public function __toString(): string
    {
        return 'I can stringz';
    }
}
```

Named Arguments

- Pass input to a method by name instead of argument's order
- More flexible code
- Better readability

```
<?php
// Using positional arguments:
array_fill(0, 100, 50);

// Using named arguments:
array_fill(start_index: 0, count: 100, value: 50);
?>
```

```
<?php
function makeyogurt($container = "bowl", $flavour = "raspberry", $style = "Greek")
{
    return "Making a $container of $flavour $style yogurt.\n";
}

echo makeyogurt(style: "natural");
?>
```

Constructor Property Promotion

```
class SomeClass
{
-   public float $someVariable;
-
    public function __construct(
-       float $someVariable = 0.0
+       public float $someVariable = 0.0
    ) {
-       $this->someVariable = $someVariable;
    }
}
```


Attributes

- Structured, machine readable metadata on declarations
- Can be inspected by ReflectionAPI

```
use Symfony\Component\Routing\Annotation\Route;

class SymfonyRoute
{
-   /**
-   * @Route("/path", name="action")
-   */
+   #[Route(path: '/path', name: 'action')]
    public function action()
    {
    }
}
```

Faul Fauler EDV-ler

The tale of the lazy IT expert



New Tools On The Block

- Automation FTW
- Silverstripe-ideannotator
- PHPCS
- PHPUnit
- PHPStan
- RectorPHP

Rector: Your Friend For Updating Code

- Much better than search/replace
- Uses Abstract Syntax Tree
- Lots of rules (rectors) for
 - upgrading language
 - code quality
 - upgrading PHPUnit
 - ...
- Possible to define own rectors for special tasks

Rector: Installation

- Via composer as dev dependency
- Big, monolithic bundle
- Totally dependency free

Create rector.php

- vendor/bin/rector init
- Configure it for your needs
 - Your code folders
 - Rules to be applied
 - Easier config: SetLists or LevelSetLists



Rector And Silverstripe CMS

Rector And Silverstripe CMS

- silverstripe-rector
- Rectors and configurations for handling Silverstripe upgrades
- Work in progress

silverstripe-rector: So Far

- Add table-name to DataObjects if missing
- Use Foo::create() instead of new Foo()

```
class SomeClass extends \SilverStripe\ORM\DataObject
{
+ private static $table_name = 'SomeClass';
+
    private static $db = [];
}
```

```
class SomeClass
{
    public function run()
    {
-     new InjectableClass($name);
+     InjectableClass::create($name);
    }
}
```

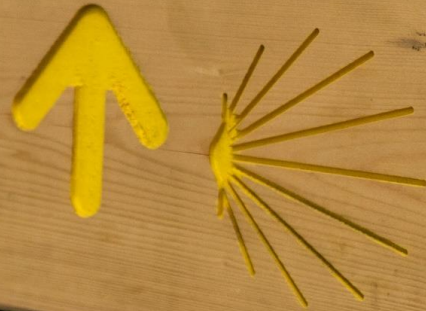
silverstripe-rector: Satisfy PHPStan

- Change `->owner` to `->getOwner()`
- Add `@config` to configuration properties

```
class SomeClass extends \SilverStripe\ORM\DataObject
{
+   /**
+    * @config
+    */
    private static $db = [];
}
```

Let's Go!

$2^8 + 2\text{km}$



258 Km






silverstripe-rector: A Possible Future

- Silverstripe deprecations
- Module deprecations
- Updated Silverstripe code style
 - E.g. use arrays for sorting or filtering DataLists
- Help needed to keep the rules up to date



Rules And More

Add Rule To silverstripe-rector

- Have an idea 
- Write a test 
- Configure rule 
- Add to Set 
- Party 

The Future Is Here



The Future Is Automated



Links, Sources, etc...

- Links:
 - Rector: <https://getrector.com/>
 - silverstripe-rector: <https://github.com/wernerkrauss/silverstripe-rector/>
- Screenshots / Code Examples:
 - Rector Documentation
 - PHP Documentation
- Images:
 - Werner Krauß, Caminho Portuguese 2023