

PHP 8.4 Обзор

Кочетов
Даниил

О чем поговорим?

*php*8.4

- Новый функционал
- Производительность
- Обратная совместимость
- Переход

Новый функционал

Функциональный стиль вызова **new**

php8.4

// Старый стиль (с дополнительными скобками):

```
$area = (new Rectangle(5, 10))->calculateArea();
```

// Новый функциональный стиль в PHP 8.4:

```
$area = new Rectangle(5, 10)->calculateArea();
```

Хуки свойств

```
class Person
{
    public string $fullName {
        get => $this->firstName . ' ' . $this->lastName;
    }

    public string $firstName {
        set => mb_ucfirst(strtolower($value));
    }

    public string $lastName {
        set {
            if (strlen($value) < 2) {
                throw new \InvalidArgumentException('Слишком короткая фамилия');
            }

            $this->lastName = $value;
        }
    }
}
```

```
$p = new Person();  
$p->firstName = 'пётр';  
echo $p->firstName; // Выведет "Пётр"
```

```
$p->lastName = 'Петров';  
echo $p->fullName; // Выведет "Пётр Петров"
```

Ассиметричный модификатор доступа

```
class Software
{
    public private(set) string $version = '8.4';
}

$software = new Software();

echo $software->version; // Выведет "8.4"

// $software->version = '8.5'; // Ошибка: нельзя изменять значение
```

Комбинирование хуков и асимметричного модификатора доступа

```
class Person
{
    private(set) string $name
    {
        get => $this->name = ucfirst(strtolower(trim($value)));
    }
}
```


Ассиметричный модификатор доступа

- **public private(set)** // Можно
- **public protected(set)** // Можно
- **protected private(set)** // Можно
- **protected public(set)** // Нельзя
- **private public(set)** // Нельзя
- **private protected(set)** // Нельзя

Функции для работы с массивами

array_find()

Функция `array_find()` находит первый элемент массива, который удовлетворяет условию.

```
$numbers = [1, 2, 3, 4, 5];  
$found = array_find($numbers, fn($n) => $n > 3);  
echo $found; // 4
```

array_find_key()

Функция `array_find_key()` находит первый ключ массива, который удовлетворяет условию.

```
$numbers = ['a' => 1, 'b' => 2, 'c' => 3];  
$key = array_find_key($numbers, fn($n) => $n === 2);  
echo $key; // 'b'
```

array_any()

Функция `array_any()` проверяет, существует ли хотя бы один элемент, который удовлетворяет условию.

```
$numbers = [1, 2, 3, 4, 5];  
$exists = array_any($numbers, fn($n) => $n > 3);  
echo $exists ? 'True' : 'False'; // True
```

array_all()

Функция `array_all()` проверяет, удовлетворяют ли все элементы массива условию.

```
$numbers = [1, 2, 3, 4, 5];  
$all = array_all($numbers, fn($n) => $n > 0);  
echo $all ? 'True' : 'False'; // True
```

Атрибут #[Deprecated]

```
#[\Deprecated(message: "use safe_replacement() instead", since: "1.5")]  
function unsafe_function()  
{  
    echo "This is unsafe", PHP_EOL;  
}  
  
unsafe_function();
```

Результат:

```
Deprecated: Function unsafe_function() is deprecated since 1.5,  
use safe_replacement() instead in example.php on line 9  
This is unsafe
```

Ленивая инициализация объектов

```
class Example
{
    public function __construct(private int $data) {}

    // ...
}

$initializer = static function (Example $ghost): void {
    $data = ...; // Получение данных или зависимостей

    $ghost->__construct($data); // Инициализация объекта
};

$reflector = new ReflectionClass(Example::class);
$object = $reflector->newLazyGhost($initializer);
```


- Поддержка работы с графемами, функция `grapheme_str_split`
- Новые функции: `mb_trim`, `mb_ucfirst`, `mb_lcfirst`
- Новая функция: `request_parse_body`
- Новые функции: `http_get_last_response_headers`,
`http_clear_last_response_headers`, `fpow`

Изменения, которые ломают обратную

`exit` и `die` : новое поведение

php8.4

```
declare(strict_types=1);  
exit(3.14);
```

PHP Fatal error: Uncaught TypeError: exit(): Argument #1 (\$status) must be of type string|int, float given

Типы параметров, неявно допускающие значение null объявлены устаревшими

php8.4

```
function greet(string $name = null): void
{
    echo "Hello, " . ($name ?? "Guest") . "!\n";
}
```

```
greet(); // Deprecated: Implicitly allowing null for parameter $name
```

Типы параметров, неявно допускающие значение null объявлены устаревшими

php8.4

```
function greet(?string $name = null): void
{
    echo "Hello, " . ($name ?? "Guest") . "!\n";
}
```

```
greet(); // Hello, Guest!
```

Обновление round

```
round(1.1, precision: 0, mode: INCORRECT_MODE); // ValueError
round(1.1, precision: 0, mode: PHP_ROUND_CEILING); // 2
round(1.1, precision: 1, mode: PHP_ROUND_FLOOR); // -1
round(1.9, precision: 0, mode: PHP_ROUND_TOWARD_ZERO); // 1
round(1.9, precision: 0, mode: PHP_ROUND_AWAY_FROM_ZERO); // 2
```

Обновления `opcache` и `JIT`

*php*8.4

По умолчанию до 8.4:

```
opcache.jit_buffer_size=0
```

```
opcache.jit=tracing
```

По умолчанию в 8.4:

```
opcache.jit_buffer_size=64M
```

```
opcache.jit=disable
```

PHP Performance Test

PHP Version: 8.3.14

JIT Enabled: No

Math Benchmark: 573.10 ms (average over 10 runs)
Array Benchmark: 101.48 ms (average over 10 runs)
String Benchmark: 452.48 ms (average over 10 runs)

PHP Performance Test

PHP Version: 8.3.14

JIT Enabled: Yes

Math Benchmark: 390.02 ms (average over 10 runs)
Array Benchmark: 88.64 ms (average over 10 runs)
String Benchmark: 439.26 ms (average over 10 runs)

PHP Performance Test

PHP Version: 8.4.1

JIT Enabled: No

Math Benchmark: 522.80 ms (average over 10 runs)
Array Benchmark: 101.61 ms (average over 10 runs)
String Benchmark: 453.33 ms (average over 10 runs)

PHP Performance Test

PHP Version: 8.4.1

JIT Enabled: Yes

Math Benchmark: 367.84 ms (average over 10 runs)
Array Benchmark: 91.38 ms (average over 10 runs)
String Benchmark: 421.07 ms (average over 10 runs)

Переход на 8.4, Ректор



```
$rectorConfig->rules([  
    ExplicitNullableParamTypeRector::class,  
    RoundingModeEnumRector::class  
]);
```

Выводы

Вопросы