# الإضافات والمميزات الجديدة لـ php 8.0

**د /إبراهيم الشامي**

عبدالمنعم فهد أحمد السوادي

2024 - 2025

## ( Union Types) - 1

هذه الميزة تسمح بإضافة أكثر من نوع للمتغير او الوسيط في الدوال

```
function getValue(int|string $value): string {
    return (string)$value;
}
echo getValue(42);
echo getValue("hello");
```

## (Nullsafe Operator)- 2

هذه الميزة تسهل التعامل مع القيم التي قد تكون null دون الحاجة للتحقق يدويا وذلك بإضافة ؟ بعد المتغير الذي نريد ان نختبر حالته

```
$user = null;
echo $user?->profile?->name ?? 'Guest';
```

## (Static Return Types) - 3

يمكن استخدام الكلمة المحجوزة static كنوع ارجاع في الدوال

```
function getStaticInstance(): static {
    static $instance = null;
    if ($instance === null) {
        $instance = new class {
            public $value = "This is a static instance";
        };
    }
    return $instance;
}

$obj = getStaticInstance();
echo $obj->value; // النتيجة: This is a static instance
```

## (Match Expression) - 4

بديل اسرع واسهل لـ switch

```
<?php
$food = 'cake';

$return_value = match ($food) {
'apple' => 'This food is an apple',
'bar' => 'This food is a bar',
'cake' => 'This food is a cake',
};

var_dump($return_value);
?>
```

## (Constructor Property Promotion) - 5

تسهل تعريف الخصائص في القوسين مباشرة دون الحاجة الى كود اخر لتعريف الخصائص
مما يقلل الكود

```
class User {
    public function __construct(
        private string $name,
        private int $age
    ) {}

    public function getInfo() {
        return "$this->name, $this->age";
    }
}

$user = new User("Ahmed", 30);
echo $user->getInfo(); // النتيجة: Ahmed, 30
```

## (Named Arguments) - 6

تمرير القيم الى الدوال باستخدام أسماء المتغيرات بدلا من الترتيب في ارسال القيم

```
function greet(string $name, string $message) {
    echo "$message, $name!";
}
greet(message: "Hello", name: "Ali"); // النتيجة: Hello, Ali!
```

## 7 ـ دوال جديدة للسلاسل النصية

نذكر بعض الدوال التي تبسط التعامل مع النصوص

```
echo str_contains('Hello World', 'World'); // النتيجة: 1 (true)
echo str_starts_with('Hello World', 'Hello'); // النتيجة: 1 (true)
echo str_ends_with('Hello World', 'World'); // النتيجة: 1 (true)
```

## (Attributes) - 8

تُستخدم لتحديد البيانات الوصفية (Metadata) للخصائص أو الدوال أو الصفوف. هذه الميزة تُعتبر بديلاً أكثر قوة وأمانًا للتعليقات التوضيحية (Annotations) التي كانت تُستخدم عبر التعليقات التقليدية.

```
#[Route('/home')]
class HomeController {
    #[Route('/index')]
    public function index() {
        echo "Welcome to Home Index!";
    }
}
```

في هذا المثال:

- الإضافة `#[Route('/home')]` تعني أن الصف `HomeController` مرتبط بالمسار `/home`.
- الإضافة `#[Route('/index')]` تعني أن الدالة `index` مرتبطة بالمسار `/index`.

- **بدون الخاصية**: يتم استخدام التعليقات التقليدية لتحديد البيانات الوصفية:

```
/**
 * @Route("/home")
 */
class HomeController {
    /**
     * @Route("/index")
     */
    public function index() {
        echo "Welcome to Home Index!";
    }
}
```

### 9 - (Spread Operator for Arrays)

يسهل دمج المصفوفات او القيم الى داخل مصفوفة أخرى

```
$arr1 = [1, 2, 3];

$arr2 = [4, 5, ...$arr1];

print_r($arr2); // النتيجة: [4, 5, 1, 2, 3]
```

### 10 - تحسينات في الأداء

تحسين كبير في أداء PHP من خلال ترجمة الشيفرة البرمجية بشكل فوري(Just-In-Time) .
تُستخدم تلقائيًا لتحسين الكود المكثف بالحسابات، مثل العمليات الرياضية

حيث ان هذه الميزة تعمل تلقائيا في الخلفية ولا تحتاج الى كود للتوضيح

### 11 - تحسينات في رسائل الخطأ

بحيث ان رسائل الخطا تظهر بشكل أوضح

```
$value = $arr['key']; // إذا لم توجد المفتاح، تظهر رسالة خطأ واضحة
```

هذا ما تم الاطلاع عليه وتدوينة والله ولي التوفيق...