الجمهورية اليمنية
الجامعة الوطنية
كلية العلوم والهندسة

PHP

المُهندسة:
ذكرى عادل الحبابي

إشراف الدكتور:
إبراهيم الشامي

❖ مقدمة:

لغة PHP تعتبر واحدة من أكثر لغات البرمجة استخدامًا لتطوير تطبيقات الويب، وقد شهدت تطورًا مستمرًا عبر العديد من الإصدارات منذ نشأتها. في هذا البحث، سنستعرض أهم الاختلافات بين PHP 8 والنسخ السابقة لها مثل PHP 7.x و PHP 5.x مع التركيز على التحسينات والميزات الجديدة في PHP 8.

اولًا: لمحة تاريخية عن الإصدارات السابقة من PHP:

- **PHP 5.x:**

تم إصدار PHP 5 في عام 2004، وكان يمثل نقلة نوعية في تطوير اللغة.
قدّم PHP 5 العديد من المزايا التي جعلت اللغة أكثر قوة ومرونة:

- **دعم الكائنات (OOP):**

تم تحسين مفهوم البرمجة الشيئية بشكل كبير، مما سمح باستخدام الوراثة، الواجهات (interfaces)، والمُنفذين(abstract classes)

- **التعامل مع قواعد البيانات:**

تم إضافة PDO (PHP Data Objects) لدعم قواعد البيانات بشكل موحد.

- **الاستثناءات:**

تم تقديم آلية التعامل مع الاستثناءات try-catch بشكل رسمي.

PHP 5.xكانت نسخة أساسية في رحلة PHP نحو تحسين الدعم للأطر الحديثة مثل Laravel و Symfony.

- **PHP 7.x:**

في عام 2015، وهو كان بمثابة إصلاح كبير للغة، مع التركيز على الأداء وتحسينات في الأنواع بالإضافة إلى العديد من التحسينات الأخرى:

- **أداء محسّن:**

PHP 7 شهد تحسينًا كبيرًا في الأداء بفضل تحسين محرك Zend وظهور محرك Zend الجديد (Zend Engine 3).

- **أنواع البيانات:**

دعم الأنواع بشكل أفضل، مثل أنواع البيانات الصارمة(strict typing)

- **تحسينات في الذاكرة:**

قللت PHP 7 من استهلاك الذاكرة بشكل كبير.

- **الاستثناءات:**
  إدخال Throwable في PHP 7 لتوحيد الاستثناءات والأخطاء.

- **حذف العديد من المزايا القديمة:**
  تم التخلص من بعض الدوال والوظائف القديمة مثل mysql_*

ثانيًا: أهم التحسينات والميزات الجديدة في PHP 8:

1. الأداء وتحسينات محرك JIT :

JIT (Just In Time Compiler):
إحدى الميزات الأبرز في PHP 8 هي JIT(المترجم في الوقت الحقيقي). يقوم JIT بتحويل بعض أجزاء من الشيفرة إلى كود machine code أثناء تنفيذ البرنامج، مما يعزز الأداء بشكل كبير، خاصةً في التطبيقات التي تتطلب حسابات معقدة أو متوازيّة.

- **التحسينات في الأداء: يمكن لـ JIT تسريع العمليات الحسابية المكثفة في بعض السيناريوهات.**

- **التأثير على التطبيقات: في بعض الحالات، قد لا يكون لـ JIT تأثير كبير على تطبيقات الويب الاعتيادية التي تعتمد على إدخال/إخراج قواعد البيانات، ولكن يمكن أن يكون مفيدًا في تطبيقات الواجهات الرسومية أو الألعاب أو عمليات التحليل المعقدة.**

2. الأنواع المتقدمة(Type System):

- **Union Types:**

أصبح بإمكان المطورين الآن تحديد أكثر من نوع للمتغير باستخدام | على سبيل المثال:

```php
<?php
function foo(int|string $value) {
    if (is_int($value)) {
        echo "القيمة هي عدد صحيح: $value\n";
    } elseif (is_string($value)) {
        echo "القيمة هي نص: $value\n";
    } else {
        echo "القيمة ليست صحيحة أو نص.\n";
    }
}

// أمثلة على استدعاء الدالة
foo(10); // ستطبع: القيمة هي عدد صحيح: 10
foo("مرحبا"); // ستطبع: القيمة هي نص: مرحبا
?>
```

- **Named Arguments:**

تم تقديم ميزة المعاملات المسماة التي تسهل استدعاء الدوال. يمكن الآن تحديد المعاملات باستخدام أسمائها بدلاً من ترتيبها:

```php
<?php
function example($name, $age) {
    echo "الاسم: $name\n";
    echo "العمر: $age\n";
}

// استدعاء الدالة مع المعاملات المسماة
example(name: "John", age: 30);
?>
```

- **Attributes (Annotations):**

بديل أكثر تطورًا لـDocBlocks ، حيث يمكن الآن إضافة معلومات إضافية إلى الدوال أو الفئات باستخدام السمة (attribute) مباشرة:

```php
<?php

use Symfony\Component\Routing\Annotation\Route;

class HomeController {

    /**
     * @Route("/home", name="home")
     */
    public function index() {
        // معالجة الطلبات التي تأتي إلى "/home"
        return "Welcome to the Home Page!";
    }
}
```

- **Constructor Property Promotion:**

تمكن هذه الميزة من إنشاء خصائص مباشرة داخل المُنشئ (constructor) باستخدام خاصية مختصرة:

```php
<?php

class Person {
    // باستخدام constructor property promotion
    public function __construct(
        public string $name,
        public int $age
    ) {
        // إذا لزم الأمر يمكن إضافة عمليات إضافية في الـ constructor
    }

    // يمكنك إضافة دوال أخرى هنا مثل دالة لعرض التفاصيل
    public function introduce(): string {
        return "مرحبًا، اسمي $this->name وأنا في $this->age سنة.";
    }
}

// إنشاء كائن من الفئة Person
$person = new Person(name: "John", age: 30);

// طباعة تعريف الشخص
echo $person->introduce();  // ستطبع: مرحبًا، اسمي John وأنا في 30 سنة.
```

3. Match Expression:

أصبح بإمكان المطورين استخدام match بدلًا من switch في بعض الحالات. الميزة توفر مقارنة صارمة ودعمًا أفضل لأنماط القيم:

```php
<?php

$input = 2;

$result = match($input) {
    1 => 'one',
    2 => 'two',
    default => 'unknown',
};

echo $result;  // ستطبع: two

?>
```

4 .تحسينات في التعامل مع الأخطاء:

تم تحسين كيفية التعامل مع الأخطاء في PHP 8 :

- **التعامل مع الأخطاء كـ Throwable: تم دمج جميع الأخطاء في فئة Throwable، مما يسمح بمعالجة الأخطاء في مكان واحد.**
- **الإعلام بشكل أفضل عن الأخطاء: اصبحت الرسائل التحذيرية والأخطاء أكثر وضوحًا، وتظهر بشكل أكثر تفصيلًا، مما يساعد في تصحيح الأخطاء بسرعة أكبر.**

5. تحسينات في الوظائف الأصلية للمكتبة:

الدوال المساعدة الجدد. تم إضافة العديد من الدوال الجديدة إلى PHP 8 ، مثل:

- **`str_contains()` : للتحقق ما إذا كانت السلسلة تحتوي على قيمة معينة.**

```php
<?php
$string = "Hello, World!";
$result = str_contains($string, "World");  // تعيد true
echo $result ? "Yes" : "No";  // ستطبع: Yes
?>
```

- **`str_starts_with()`: للتحقق ما إذا كانت السلسلة تبدأ بقيمة معينة.**

```php
<?php
$string = "Hello, World!";
$result = str_starts_with($string, "Hello");  // تعيد true
echo $result ? "Yes" : "No";  // ستطبع: Yes
?>
```

- **`str_ends_with()`:للتحقق ما إذا كانت السلسلة تنتهي بقيمة معينة.**

```php
<?php
$string = "Hello, World!";
$result = str_ends_with($string, "World!");  // تعيد true
echo $result ? "Yes" : "No";  // ستطبع: Yes
?>
```

- **`get_debug_type()`: تعيد نوع المتغير كما هو مخصص للعرض في حالة التصحيح**

```php
<?php
$value = 42;
echo get_debug_type($value);  // ستطبع: int

$array = [1, 2, 3];
echo get_debug_type($array);  // ستطبع: array
?>
```

- **مكتبة `fdiv()`:**

تمت إضافة دالة fdiv() لتمكين العمليات الحسابية التي تتضمن القسمة على صفر بشكل صحيح، مما يتيح التحكم بشكل أفضل في الأخطاء الحسابية.

```php
<?php
$result = fdiv(10, 2);  // تعيد 5
echo $result;

$result = fdiv(10, 0);  // تعيد INF
echo $result;

$result = fdiv(0, 0);   // تعيد NAN
echo $result;
?>
```

6. التوافق العكسي(Backward Compatibility):

- **إزالة بعض الوظائف القديمة: تم إزالة بعض الوظائف التي كانت موجودة في الإصدارات القديمة من PHP مثل each() و create_function() وغيرها.**

- **التغييرات في بعض السلوكيات: بعض السلوكيات القديمة تم تغييرها أو تحسينها، مثل filter_var() و strstr() وغيرها.**

```php
<?php
$email = "user@domain.com";
if (filter_var($email, FILTER_VALIDATE_EMAIL)) {
    echo "البريد الإلكتروني صحيح.";
} else {
    echo "البريد الإلكتروني غير صحيح.";
}
?>
```

```php
<?php
$string = "Hello, World!";
$result = strstr($string, "World", true);  // تعيد "Hello, "
echo $result;
?>
```

7. تحسينات في الأمان:

- تم تحسين التعامل مع البيانات المدخلة للحد من الثغرات الأمنية مثل SQL Injection أو XSS.
- تم تعزيز سياسات الأمان حول كلمات المرور وعملية التحقق من صحة المدخلات.

ثالثًا: مقارنة بين PHP 8 والنسخ السابقة:

الفرق في الأداء:

- **PHP 7.x:** تقدم تحسينات كبيرة في الأداء مقارنة بـ PHP 5.x ، ولكن PHP 8 تقدم تحسينات أكثر في بعض التطبيقات باستخدام JIT .
- **PHP 8:** مع إضافة JIT ، تحسّن الأداء في تطبيقات معينة بشكل ملحوظ مقارنة بـ PH 7.x

الفرق في اللغة:

- **PHP 7.x:** قدم دعماً أفضل للأنواع والبرمجة الشيئية.
- **PHP 8:** يقدم مزايا جديدة مثل Union Types ، Named Arguments، Match Expressions، Attributes، وتحسينات أخرى في البرمجة الشيئية.

الفرق في التعامل مع الأخطاء:

- **PHP 7.x: قدم تحسينات في التعامل مع الاستثناءات والأخطاء.**
- **PHP 8: تعزيز كبير في كيفية معالجة الأخطاء والرسائل التحذيرية.**

الفرق في الأمان:

- **PHP 7.x: شهد تحسينات أمنية كبيرة.**
- **PHP 8:تم إضافة تحسينات أمنية إضافية، مثل التحقق الصارم من البيانات.**

❖ خاتمة:

PHP 8تقدم العديد من التحسينات والميزات التي تجعلها نسخة قوية ومحدثة مقارنة بالنسخ السابقة. من أهم التحسينات: إضافة JIT لتحسين الأداء، دعم الأنواع المتقدمة، إضافة match expressions وattributes بالإضافة إلى تحسينات في التعامل مع الأخطاء والأمان. لكن، يجب أن يكون المطورون على دراية بالتغييرات التي قد تؤثر على التوافق العكسي، خاصةً إذا كانوا يعملون مع إصدارات قديمة مثل PHP 5.x أوPHP 7.x.