|  |
| --- |
| **تطوير ويب2** |
| Value Method |
| د/إبراهيم الشامي |

|  |
| --- |
| **هديل خالد عبدالله الحزورة**  علوم حاسوب |

في لغة PHP، **القيمة** (Value) و**المرجع** (Reference) هما أسلوبان للتعامل مع البيانات داخل المتغيرات أو عند تمريرها للدوال. لفهم الفرق بينهما وكيفية التعامل مع "طرق القيمة" (Value Methods)، يجب أن نفهم الآتي:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **ما هي Value Method؟**

عند تمرير المتغيرات أو استخدامها في PHP، يمكن أن يتم التعامل مع البيانات كقيمة (Value) أو كمرجع (Reference):

1. **Value (القيمة):**
   * يتم نسخ محتوى المتغير الأصلي إلى متغير جديد.
   * أي تغيير في المتغير الجديد لا يؤثر على المتغير الأصلي.
   * يُعتبر الأسلوب الافتراضي في PHP.
2. **Reference (المرجع):**
   * يتم تمرير "مرجع" إلى المتغير الأصلي.
   * أي تعديل على المتغير الجديد يؤثر مباشرةً على المتغير الأصلي.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **Value Method في PHP**

عندما يتم تمرير المتغيرات إلى الدالة باستخدام القيمة، فإن PHP تنسخ قيمة المتغير بدلاً من تمرير المتغير نفسه. بالتالي، يتم عزل المتغيرات الأصلية عن أي تغييرات تحدث داخل الدالة.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **مثال عملي على Value Method:**

**الكود:**

function modifyValue($num) {

$num += 10;

echo "Inside the function: $num" . PHP\_EOL;

}

$originalValue = 20;

modifyValue($originalValue);

echo "Outside the function: $originalValue" . PHP\_EOL;

**الإخراج:**

Inside the function: 30

Outside the function: 20

**الشرح:**

* داخل الدالة modifyValue، يتم نسخ القيمة الأصلية $originalValue إلى المتغير $num.
* التعديل داخل الدالة لا يؤثر على المتغير الأصلي لأن المتغير الجديد يحتوي على نسخة منفصلة من القيمة.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **تأثير Value Method على المصفوفات**

**الكود:**

function addElement($arr) {

$arr[] = "New Element";

print\_r($arr);}

$originalArray = [1, 2, 3];

addElement($originalArray);

print\_r($originalArray);

**الإخراج:**

Array ( [0] => 1 [1] => 2 [2] => 3 [3] => New Element )

Array ( [0] => 1 [1] => 2 [2] => 3 )

**الشرح:**

* يتم نسخ المصفوفة الأصلية $originalArray إلى المتغير $arr.
* التعديل على $arr داخل الدالة لا ينعكس على $originalArray.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **كيف يختلف عن Reference Method؟**

**Reference Method باستخدام الرمز &:**

function modifyByReference(&$num) {

$num += 10;

echo "Inside the function: $num" . PHP\_EOL;

}

$originalValue = 20;

modifyByReference($originalValue);

echo "Outside the function: $originalValue" . PHP\_EOL;

**الإخراج:**

Inside the function: 30

Outside the function: 30

**الفرق:**

* في طريقة القيمة (Value Method)، يتم التعامل مع نسخة من المتغير.
* في طريقة المرجع (Reference Method)، يتم تمرير المتغير الأصلي مباشرةً، وبالتالي تتأثر القيمة الأصلية.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **الفرق بين Value Method و Reference Method**

| **خاصية** | **Value Method** | **Reference Method** |
| --- | --- | --- |
| التعديل على القيمة | لا يؤثر على الأصل | يؤثر مباشرةً على الأصل |
| الاستخدام الافتراضي | نعم | لا |
| الأداء | أقل كفاءة عند تمرير بيانات ضخمة | أكثر كفاءة مع البيانات الكبيرة |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **متى نستخدم Value Method؟**
* عند الحاجة إلى ضمان عدم تعديل البيانات الأصلية.
* عند الرغبة في العمل على نسخة من البيانات لحماية القيم الأصلية.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **متى نستخدم Reference Method؟**
* عندما تحتاج إلى تعديل البيانات الأصلية.
* لتوفير الأداء عند العمل مع بيانات كبيرة (مثل المصفوفات الكبيرة).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_