الفصل الثامن

التعامل مع الكائنات المتجهة

التعامل مع الكائنات المتجهة:

من اهم المزايا فى لغة PHP هى امكانية استخدام الكائنات المتجهة و التى تتميز بانها لغة العصر الان لقوتها و ثباتها بالنسبة لانظمة التشغيل المختلفة و المفاهيم القادة تنطبق على كل اللغات الحديثة الموجودة الآن.

ماهو الكائن ؟

الكائن يمكنك ان تعتبره محتوى للدوال و المتغيرات المختلفة الذى ينحدر من نموذج خاص يسمى class و يقوم الكائن باخفاء عمله الداخلى عن الكود الذى يقوم باستخدامه وبدلا من ذلك يقدم واجهة سهلة للتعامل معه تسمى methods و التى بدورها لها وصول الى متغيرات خاصة تسمى properties .

تبدء البرمجة المتجهة بانشاء class او فئة لهذا الكائن و تتميز بمجموعة من السمات التى يتم توريثها للكائن بعد ذلك و لكن تختلف فى بعض خصائصها من كائن لاخر

و الفئة عبارة عن دوال خاصة تسمى methods و متغيرات خاصة تسمى properties

انشاء كائن:

لكى نستطيع انشاء كائن يجب ان ننشئ الفئة (class) المنحدر منها كما يلى:

```
class first_class
{
// وكود الفئة هنا
}
```

و يمكنك بعد ذلك ان تقوم باشاء نسخ من هذه الفئة كما يلى

```
$obj1 = new first_class();
$obj2 = new first_class();
```

```
print "\$obj1 is a ".gettype($obj1)."<br>";
print "\$obj2 is a ".gettype($obj2)."<br>";
```

لا حظ انه يجب ان تستخدم العبارة new() حتى يتم الانشاء وتقوم الدالة gettype() بارجاع نوع المتغير وفى حالتنا تقوم بارجاعة كلمة "object" التى يتم طباعتها

خصائص الكائنات:

يتم تعريف properties الخاصة بكائن عن طريق تعريف متغيرات خاصة توضع فى او تعريف الكائن كما يلى و يمكن ان تكون الخاصية عبارة عن قيمة حرفية او عددية او مصفوفة او حتى كائن اخر كما يتضح من المثال:

```
class first_class
{
var $name = "رامى";
}
```

لاحظ ان تعريف المتغيرات يتم عن طريق العبارة var والا سينتج خطأ

و الان تستطيع ان الوصول الى الخاصية name و حتى تغيير قيمتها الافتراضية كما يلى:

```
class first_class
{
var $name = "harry";
}

$obj1 = new first_class();
$obj2 = new first_class();
$obj1->name = "ماجد";
print "$obj1->name<BR>";
print "$obj2->name<BR>";
```

لاحظ انه تم تعديل الخاصية للكائن obj1 عن طريق العلامة -> التى يمكن عن طريقها الوصول الى الخصائص المتفرعة من الكائن

و يمكن استخدام الكائنات لتخزين البيانات المختلفة مثل المصفوفات ولكن بطريقة اكثر مرونة و اكثر قوة

وسائل الكائنات:

يتم توريث Object Methods من الفئات classes الابوية عند انشاء اى كائن بالعبارة new() كما يتضح من المثال:

```
1: <html dir="rtl">
2: <head>
3: <title>Object Methods</title>
4: <body>
5: <?php
6: class first_class
7: {
8: var $name;
9: function sayHello()
10: {
11: print "مرحب بك";
12: }
13: }
14:
15: $obj1 = new first_class();
16: $obj1->sayHello();
17: // يتم طبع كلمة مرحب بك
18: ?>
19: </body>
20: </html>
```

لاحظ كيف تم الوصول الى الوسيلة التى تم تعريفها فى داخل الفئة و بنفس الطريقة يمكن الوصول الى الخصائص:

```
1: <html dir="rtl">
2: <head>
3: <title>خصائص الكائنات</title>
4: <body>
5: <?php
6: class first_class
7: {
8: var $name="ماجد";
9: function sayHello()
10: {
11: print "أهلا بكم انا اسمى  $this->name<BR>";
12: }
13: }
14:
15: $obj1 = new first_class();
16: $obj1->sayHello();
17: // يتم طبع اهلا بكم انا اسمى ماجد
18: ?>
19: </body>
20: </html>
```

لاحظ استخدام العبارة this التى تشير الى الكائن الحالى و بنفس الطريقة يمكن الوصول الى المتغيرات و تغيير قيمتها داخل الوسائل المعرفة كما يلى:

```
1: <html dir="rtl">
2: <head>
3: <title>تغيير الخصائص</title>
4: </head>
5: <body>
6: <?php
131
```

```
7: class first_class
8: {
9: var $name="ماجد";
10: function setName( $n )
11: {
12: $this->name = $n;
13: }
14: function sayHello()
15: {
16: print "مرحب بك انا اسمى  $this->name<BR>";
17: }
18: }
19:
20:
21: $obj1 = new first_class();
22: $obj1->setName("اسامة");
23: $obj1->sayHello();
24: // يتم طباعة مرحب بك انا اسمى اسامة
25: ?>
26: </body>
27: </html>
```

لاحظ انه تم تغيير قيمة الخاصية name و ان الكائن قام بالتحكم فى الخاصية عن طريق تقديم الوسيلة setName التى تستقبل معامل هو الاسم تماما مثل الدوال العادية.

هناك تقنية هامة تستخدم مع الفئات وهى امكانية انشاء وسيلة منشئة اى constructor method و يتم استدعائها تلقائيا عند انشاء الكائن اذا كان لها نفس اسم الفئة و يمكن للكائنات ان تقوم باستدعاء كود فى داخلها لكى تقوم بتمهيد نفسها كما يتضح من المثال الاتى:

```
1: <html dir="rtl">
2: <head>
```

```
3: <title>الوسائل المنشئة للكائنات</title>
4: </head>
5: <body>
6: <?php
7: class first_class
8: {
9: var $name;
10: function first_class( $n="ماجد" )
11: {
12: $this->name = $n;
13: }
14: function sayHello()
15: {
16: print "مرحب بك انا اسمى  $this->name<BR>";
17: }
18: }
19: $obj1 = new first_class("اسامة");
20: $obj2 = new first_class("اشرف");
21: $obj1->sayHello();
22: // طباعة الرسالة مرحب بك انا اسمى اسامة
23: $obj2->sayHello();
24: // طباعة الرسالة مرحب بك انا اسمى اشرف
25: ?>
26: </body>
27: </html>
```

عند انشاء الكائن يتم وضع القيمة الافتراضية "ماجد" فى خاصية الاسم فأذا لم نقم بتعيين قيمة للمعامل الخاص بالعبارة new() فأن قيمة هذه الخاصية تبقى كما هى

مثال عام:
سنقوم الان بتطبيق ما تعلمناه سابقا فى انشاء جدول يمكنه التعامل مع الحقول والصفوف بحيث يمكن التحكم فى عرض البيانات على شكل سطور اسفل بعضها

تعريف الخصائص:

اول خطوة هى معرفة ما هى الخصائص الهامة التى نحتاج الى تعريفها فسنحتاج فى هذا المثال الى تعريف مصفوفة للاعمدة و مصفوفة متعددة للصفوف و ايضا قيمة رقيمة لعدد الاعمدة التى نتعامل معها كما يلى:

```
class Table
{
var $table_array = array();
var $headers = array();
var $cols;
}
```

وسيلة الانشاء:

يمكن الحصول على عدد الاعمدة من constructor method بحيث يتم ارجاع اسماء الاعمدة فى مصفوفة وبالتالى يتم حساب عدد الاعمدة ووضعها فى خاصية جديدة cols :

```
function Table( $headers )
{
$this->headers = $headers;
$this->cols = count ( $headers );
}
```

وعن طريق تخزين هذه المعلومات فى خصائص الكائن فأن كل الوسائل Methods سيكون لها امكانية الوصول الى هذه المعلومات ايضا وسنبدء الان بانشاء هذه الوسائل:

الوسيلة **addRow()**:

ستقوم هذه الوسيلة بزيادة عدد الصفوف التى هى على شكل مصفوفة مرتبة بنفس ترتيب الاعمدة وفيما يلى تعريف هذه الوسيلة

```
function addRow( $row )
{
  if ( count ($row) != $this->cols )
  return false;
  array_push($this->table_array, $row);
  return true;
}
```

تستقبل الوسيلة مصفوفة تساوى عناصرها عدد الاعمدة فى الجدول و يتم اختبار ذلك عن طريق الدالة count فأذا لم تكن متساوية فيتم ارجاع القيمة false.

تقوم الدالة array_push باضافة عناصر الى مصفوفة فأذا كان العنصر المضاف هو نفسه مصفوفة فيتم انشاء مصفوفة متعددة Multidimensional array.

أضافة الوسيلة addRowAssocArray() :
تقدم هذه الوسيلة امكانية اضافة مصفوفة مترابطة بحيث لا يشترط اضافة القيم بنفس ترتيب الاعمدة و تقبل مفاتيح عناصر المصفوفة ويتم مقارنتها بالاعمدة فأذا لم تكن متماثلة يتم تجاهل هذه القيم و يتم تعريف هذه الوسيلة كما يلى:

```
function addRowAssocArray( $row_assoc )
{
$row = array();
foreach ( $this->headers as $header )
{
if ( ! isset( $row_assoc[$header] ))
$row_assoc[$header] = "";

$row[] = $row_assoc[$header];
}
array_push($this->table_array, $row);
return true;
}
```

يتضح من المثال كيفية اختبار مفاتيح العناصر للمصفوفة المضافة فأذا لم يتم وضع قيم لها فأن الدالة تقوم بوضع لا شئ "" فى هذا الحقل او العمود

لقد قمنا الان بتعريف وسيلتين للاضافة داخل المصفوفة و يتبقى وسيلة لعرض البيانات

الوسيلة **()Output**:

تقوم هذه الوسيلة بعرض رأس الجدول و المصفوفة المخزنة فى خاصية array فى المتصفح ويتم تعريفها كمايلى:

```
function output()
{
print "<pre>";
foreach ( $this->headers as $header )
print "<B>$header</B> ";
print "\n";
foreach ( $this->table_array as $y )
{
foreach ( $y as $xcell )
print "$xcell ";
print "\n";
}
print "</pre>";
}
```

من الكود السابق يتضح انه يتم طباعة راس الجدول اولا ثم خلايا الجدول ولان هذه الخلايا عبارة عن مصفوفة ثنائية فنه يتم طباعة الجدول عن طريق حلقتين باستخدام العبارة foreach

وفيما يلى كود المثال بالكامل:

```
1: <html>
2: <head>
```

```
3: <title>مثال على الكائنات</title>
4: </head>
5: <body>
6: <?php
7: class Table
8: {
9: var $table_array = array();
10: var $headers = array();
11: var $cols;
12: function Table( $headers )
13: {
14: $this->headers = $headers;
15: $this->cols = count ( $headers );
16: }
17:
18: function addRow( $row )
19: {
20: if ( count ($row) != $this->cols )
21: return false;
22: array_push($this->table_array, $row);
23: return true;
24: }
25:
26: function addRowAssocArray( $row_assoc )
27: {
28: $row = array();
29: foreach ( $this->headers as $header )
30: {
31: if ( ! isset( $row_assoc[$header] ))
32: $row_assoc[$header] = "";
33: $row[] = $row_assoc[$header];
34: }
35: array_push($this->table_array, $row);
36: return true;
```

```
37: }
38:
39: function output()
40: {
41: print "<pre>";
42: foreach ( $this->headers as $header )
43: print "<B>$header</B> ";
44: print "\n";
45: foreach ( $this->table_array as $y )
46: {
47: foreach ( $y as $xcell )
48: print "$xcell ";
49: print "\n";
50: }
51: print "</pre>";
52: }
53: }
54:
55: $test = new table( array("a","b","c") );
56: $test->addRow( array(1,2,3) );
57: $test->addRow( array(4,5,6) );
58: $test->addRowAssocArray( array ( b=>0,
a=>6, c=>3 ) );
59: $test->output();
60: ?>
61: </body>
62: </html>
```

وتكون نتيجة تنفيذ المثال السابق هو طبع الناتج:

```
a    b    c
1    2    3
4    5    6
6    0    3
```

مما سبق تتضح فوائد استخدام الفئات او class فى البرنامج فيمكنك اعادة استعمال الكود مرة اخرى فى أى مشروع يحتاج ان يقوم باظهار المعلومات بهذه الطريقة مما يوفر الوقت و الجهد.