الفصل السادس

الدوال

دار البراء

الدوال **Functions**:

استخدام الدوال هام جدا حتى لا يكون البرنامج كبير الحجم وصعب الفهم كما تظهر اهميتها عند تكرار الحاجة لاداء نفس الوظيفة مرة اخرى .
ويمكنك ان تفكر فى الدوال مثل الالة التى يدخل اليها المادة الخام و تقوم بعد المعالجة بإخراج الناتج فمثلا اذا اردت ان تقوم بخبز كعكة مرة واحدة فأنك سوف تفعلها بنفسك ولكن اذا اردت ان تخبز الكعكة الاف المرات فأنك يجب ان تقوم بشراء مكينة لتحضير الكعك و بالمثل فأن تقرير استخدام الدوال او لا هو الحاجة لاستخدام نفس الكود العديد من المرات .
لذلك عندما تقوم بنداء دالة فأنك تعطيها قيمة او عدة قيم و تقوم هى بعد معالجة خاصة لهذه القيم بارجاع قيمة معينة يمكنك ان تقوم باستخدامها فى الكود .
والدوال تنقسم الى نوعين دوالة مبنية فى اللغة ذاتها و دوال تقوم انت نفسك بتعريفها .

مثال لدالة جاهزة **(built-in)**:

سنقوم الان باستخدام الدلة abs() التى تقوم بأخذ قيمة عشرية (رقم سالب مثلا) وتقوم الدالة بارجاع القيمة المطلقة (الصحيحة) لهذا العدد ويتم تنفيذ ذلك كما يلى:

```
1: <html dir="rtl">
2: <head>
3: <title>دالة الاعداد الصحيحة</title>
4: </head>
5: <body>
6: <?php
7: $num = – 321;
8: $newnum = abs( $num );
9: print $newnum;
10: // prints "321"
11: ?>
12: </body>
13: </html>
```

كما يتضح فان المتغير newnum سيحتوى فى السطر 8 على القيمة الصحيحة للعدد (-321)

تعريف الدوال:

الصيغة القياسية لتعريف الدوال هى:

```
function some_function( $argument1, $argument2
)
{
// function code here
}
```

حيث argument1 و argument2 هى المعاملات او المادة الخام التى نقوم بتمريرها الى الدالة ومن ثم تقوم الدالة بعد المعالجة بارجاع القيمة النهائية

مثال:

فى هذا المثال سنقوم بتعريف دالة و النداء عليها كما يلى:

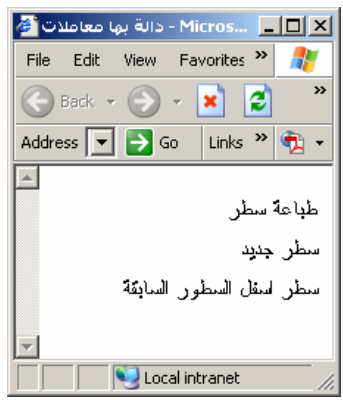
```
1: <html dir="rtl">
2: <head>
3: <title>تعريف دالة</title>
4: </head>
5: <body>
6: <?php
7: function bigtitle()
8: {
9: print "<h1>مرحب بك</h1>";
10: }
11: bigtitle();
12: ?>
13: </body>
14: </html>
```

كما سبق فى المثال فقد قمنا بتعريف الدالة وفى هذه الحالة لاتأخذ اى معاملات ولكن عند النداء عليها تقوم بطبع كلمة "مرحب بك" بخط كبير .

سنقوم الان بكتابة مثال لدالة بها معاملات:

```
1: <html dir="rtl">
2: <head>
3: <title>دالة بها معاملات</title>
4: </head>
5: <body>
6: <?php
7: function print_line_msg( $txt )
8: {
9: print ("$txt<br>\n");
10: }
11: print_line_msg ("طباعة سطر");
12: print_line_msg ("سطر جديد");
13: print_line_msg ("سطر اسفل السطور السابقة");
14: ?>
15: </body>
16: </html>
```

و يجب ان تكون نتيجة تنفيذ السطور السابقة هى الشكل:



نلاحظ من هذا المثال اننا قمنا باعطاء الدالة قيم مختلفة وفى كل مرة تقوم الدالة بطباعة النص و ادراج سطر جديد اسفل النص عن طريق كود HTML ""
 /n

سنقوم الان بتعريف دالة تقوم بارجاع قيمة معينة:

```
1: <html>
2: <head>
3: <title>دالة تعطى قيمة</title>
4: </head>
5: <body>
6: <?php
7: function addNums( $firstnum, $secondnum;
8: {
9: $result = $firstnum + $secondnum )
10: return $result;
11: }
12: print addNums(3,5);
13: // 8 النتيجة ستكون
14: ?>
15: </body>
16: </html>
```

من المثال البسيط السابق نرى ان الدالة addNum تقوم باضافة رقم اول معامل الى رقم ثانى معامل وتقوم بطبع الناتج و يتم تحقيق ذلك فى السطر رقم 10 الذى يحتوى على العبارة return والتى تقوم بتخزين القيمة للمتغير result فى الدالة .

مدى المتغيرات:

المتغيرات المعلن عنها داخل الدوال تبقى معرفة فقط داخل حدود الدالة ولا يمكن الوصول اليها خارجها وهذا مفيد لانه يحميك من ان تقوم بتغيير قيمة متغير على سبيل الخطأ خارج الدالة .

والمثال الاتى يقوم بتوضيح عدم امكانية الوصول الى المتغير المحلى المعرف داخل الدالة:

```
1: <html dir="rtl">
2: <head>
3: <title>مدى المتغير المحلى</title>
4: </head>
5: <body>
6: <?php
7: function test()
8: {
9: $testvariable = "this is a test variable";
10: }
11: print "test variable: $testvariable<br>";
12: ?>
13: </body>
14: </html>
```

نرى فى هذا المثال ان برنامج IE لا يقوم بطباعة شيئ لان المتغير نفسه غير موجود خارج الدالة .

تعريف المتغيرات العالمية **Global Variables**:

بعكس المتغيرات المحلية يمكن الوصول الى المتغيرات العالمية من اى مكان فى البرنامج .

مثال:

```
1: <html dir="rtl">
2: <head>
3: <title>مدى المتغيرات</title>
4: </head>
5: <body>
```

```
6: <?php
7: $life = 42;
8: function meaningOfLife()
9: {
10: print "The meaning of life is $life<br>";
11: }
12: meaningOfLife();
13: ?>
14: </body>
15: </html>
```

تكون نتيجة المثال السابق طباعة لاشئ لان المتغير life ليس معرفا داخل الدالة فلا تكون له القيمة 42 ولكن لكى تستطيع ان تصل الى قيمة من خارج الدالة يجب تمريرها عن طريق المعاملات .

ويمكن ايضا الوصول الى متغيرات الخارجية ولكن يجب فى هذه الحالة استخدام العبارة global كما يلى:

```
1: <html>
2: <head>
3: <title>Listing 6.8</title>
4: </head>
5: <body>
6: <?php
7: $life=42;
8: meaningOfLife()
9: {
10: global $life;
11: print "The meaning of life is $life<br>";
12: }
13: meaningOfLife();
14: ?>
15: </body>
16: </html>
```

نرى من المثال السابق ان المتصفح يقوم بطباعة قيمة المتغير life لانه تم الاعلان عنه بالعبارة global و تكون النتيجة هى طباعة الرسالة:

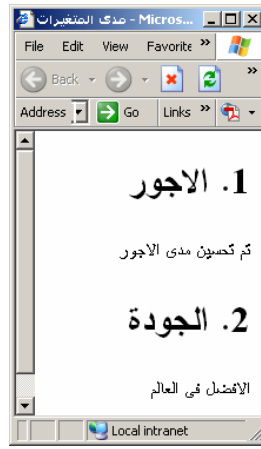
The meaning of life is 42

ذكرنا ان المتغير المحلى المعرف داخل الدالة يتم استخدامه ثم تنتهى حياته عند انتهاء تنفيذ الدالة ولاكن يمكن ايضا الحفاظ على قيمة المتغير داخل و خارج الدالة بحيث اذا تم تنفيذ الدالة مرة اخرى يتم تذكر اخر قيمة للمتغير و يتم ذلك بطريقتين أما باستخدام العبارة global او باستخدام العبارة static .

فيما يلى كيفية اداء ذلك عن طريق العبارة global:

```
1: <html dir="rtl">
2: <head>
3: <title>مدى المتغيرات</title>
4: </head>
5: <body>
6: <?php
7: $num_of_calls = 0;
8: function second_item( $txt )
9: {
10: global $num_of_calls;
11: $num_of_calls++;
12: print "<h1>$num_of_calls. $txt</h1>";
13: }
14: second_item("الاجور");
15: print("تم تحسين مدى الاجور<p>");
16: second_item ("الجودة");
17: print("الافضل فى العالم<p>");
18: ?>
19: </body>
20: </html>
```

وتكون نتيجة المثال الشكل الآتى:



و يمكن تنفيذ ماسبق عن طريق العبارة static كما يلى:

```
1: <html dir="rtl">
2: <head>
3: <title>Static</title>
4: </head>
5: <body>
6: <?php
7: function second_item ( $txt )
8: {
9: static $num_of_calls = 0;
10: $num_of_calls++;
100
11: print "<h1>$num_of_calls. $txt</h1>";
12: }
13: second_item ("الاجور ");
14: print("تم تحسين مدى الاجور <p>");
15: second_item ("الجودة ");
16: print("الافضل فى العالم <p>");
```

```
17: ?>
18: </body>
19: </html>
```

وستكون النتيجة مثل المثال السابق تماما ولكن باسلوب افضل

الحاق المعامل بقيمة افتراضية:

يمكنك ان تقوم بوضع قيمة افتراضية للمعامل الذى يتم تمريره الى الدالة بحيث لا تحتاج الى وضعه ضمن معاملات الدالة عند النداء عليها .

مثال:

```
1: <html dir="rtl">
2: <head>
3: <title>معاملات الدوال</title>
4: </head>
5: <body>
6: <?php
7: function fontWrap( $txt, $size=3 )
8: {
9: print "<font
size=\"$size\"face=\"Helvetica,Arial,Sans-
Serif\">$txt</font>";
10: }
11: fontWrap("عنوان كبير<br>",5);
12: fontWrap("حروف نصية<br>");
13: fontWrap("المزيد من الحروف النصية<br>");
14: fontWrap("حروف نصية اخرى<br>");
15: ?>
16: </body>
17: </html>
```

يمكنك ان ترى من المثال السابق اننا لم نحتاج الى الحاق قيمة للمعامل size ويتم استخدام القيمة الافتراضية 3 فى هذه الحالة وتكون النتيجة كما بالشكل:

تمرير المتغيرات المرجعية:

افتراضيا فى الحالات العادية يتم تمرير المتغيرات فى معاملات الدوال عن طريق قيمة المتغير اى ان اى تغيير فيى قيمة المتغير داخل الدالة لا يؤثر على قيمة المتغير خارجها وهذا يسمى التمرير بالقيمة او (variable by value) اما اذا اردنا ان نتحكم فى قيمة المتغير و نقوم بتغييرها حتى بعد انتهاء عمل الدالة يجب ان نستخدم هذه التقنية التى تسمى (variable by reference) ويتم ذلك كما بالمثال:.

```
1: <html dir="rtl">
2: <head>
3: <title>variables by reference</title>
4: </head>
5: <body>
6: <?php
7: function addFive( &$num )
8: {
9: $num += 5;
10: }
11: $orignum = 10;
12: addFive( $orignum );
```

```
13: print ( $orignum );
14: ?>
15: </body>
16: </html>
```

هل تستطيع الان ان تخمن ماهى نتيجة المثال السابق ؟ سيتم ببساطة تغيير قيمة المتغير orignum فى السطر 12 وسيتم طبع النتيجة 15 فى المتصفح.