الفصل الثانى

قواعد لغة PHP

قواعد لغة PHP:

فيما يلى سنقوم بشرح لاهم قواعد لغة PHP و كيفية كتابة الاوامر .

التعليقات:

هى مجرد سطور تشرح المغزى من جزء معين من الكود و تظهر فائدتها عند كتابة برنامج كبير وتعديله بعد مرور سنوات فيصبح من العسير جدا على المطور تذكر فائدة السطور التى كتبها او المخرجات التى ستنتج عنها لذلك يتم ادراج التعليقات كشرح للمطور ليس اكثر و لا يقوم المعالج الخاص باللغة بترجمة هذه التعليقات او الالتفات اليها .

والسؤال الان كيف يعرف المترجم سطور التعليقات من سطور الكود ؟

يتم ذلك عن طريق علامات خاصة و توجد طريقتين الاولى يمكن فيها ادراج تعليق بعد العلامة // حتى وان كان يسبقها سطر كود عادى

مثال:

```
<?
echo "hi..."; //هذا سطر تعليق
?>
```

والطريقة الثانية تستخدم لادراج التعليق على اكثر من سطر عن طريق علامة /* هكذا :

```
<?
  /* سطر تعليق 1
  */ سطر تعليق 2
   echo "hi...";
?>
```

المتغيرات:

هى من اساسيات اى لغة و تستخدم لتخزين قيمة معينة فى ذاكرة الكمبيوتر و يتم تعريف المتغير عن طريق علامة الدولار "$" بالصيغة الاتية:

$اسم_المتغير=قيمة المتغير;

نلاحظ من الصيغة السابقة انه يتم اختيار الاسم الذى نريده للمتغير (يجب ان يكون بالانجليزية) ووضع العلامة $ قبله ثم نكتب علامة التساوى التى تقوم بتخزين القيمة فى المتغيرو اخيرا الفاصلة المنقوطة اخر اى سطر من سطور لغة PHP .

قيمة المتغير قد تكون نصية او عددية فى المثال التالى سنقوم بتخزين قيمة نصية فى متغير اسمه name:

```
$name="بخيت";
```

لغة PHP حساسة لحالة الاحرف فمثلا لا يمكن تخصيص قيمة للمتغير name ثم فحص قيمتها بالنداء على المتغير Name فى هذه الحالة يكون الاسمين متغيرين منفصلين تماما .

مثال:

```
$name="رامى";
$Name="ايهاب";
echo $name;
```

هل تستطيع توقع اى الاسمين سيتم طباعته ؟

وكقاعدة عامة لتسهيل عملية تطوير البرامج و لكى تستطيع تذكر الهدف من المتغير بسهولة يجب ان يعبر اسم المتغير عن وظيفته ولايسمح بمسافات خالية فى اسم المتغير بل يجب ان يكون متصل لذلك تستطيع استخدام علامة "_" بين الكلمات فمثلا اسم متغير لتخزين درجة الحرارة يكون نموذجيا بالاسم $heat_degree وهكذا .

نلاحظ من الامثلة السابقة اننا قمنا بتخزين قيم نصية او حرفية ويتم التعبير عنها بين علامتى التنصيص "" و هى قيم لا يمكن اجراء عمليات حسابية عليها اما اذا اردنا تخزين قيم عددية يمكن اجراء عمليات حسابيه لها يجب ان نتعرف اولا على انواع البيانات الممكن استخدامها فى لغة PHP .

انواع البيانات:

1- بيانات حرفية:

يعتبر معالج لغة PHP اى قيمة موضوعة بين علامتى التنصيص مفردة او مزدوجة على انها قيمة نصية وفيما يلى امثله لذلك:

```
$var1="text";
$var2='some string...';
```

واذا اردت ادراج العلامة ' فيجب كتابة النص بين العلامتين "" اما اذا اردت كتابة العلامة" داخل النص فيجب كتابتها بعد العلامة \ كما يلى:

```
$str_var="my name is \"bekheet\""
```

اذا اردت طباعة مسار ملف مثلا بحيث يتم وضع العلامتين \\ بجانب بعضهم فى هذه الحالة يجب كتابة القيمة كمايلى:

```
$v="c:\\\windwos\\\system32";
```

وتكون نتيجة طباعة هذا السطر هى

```
c:\\windows\\system32
```

وفى حالة اذا قمنا بتخزين رقم الشارع فى متغير و اسم المدينة فى متغير اخر فلكى نستطيع طباعة العنوان كاملا يجب ان نقوم بربط المتغيرين فى متغير جديد كما يلى:

```
$v_st="306 st,";
$v_city="Alexandria";
$full_Address=v_st.' '.v_city;
```

وتكون النتيجة هى السطر:

```
306 st, Alexandria
```

لاحظ هنا اننا قمنا بالربط عن طريق علامة النقطة (.) و قمنا باضافة مسافة خالية حتى لايكون الكلام ملتصق ببعض .

ملحوظة: يمكنك عند اختبار الامثلة فتح نسخة واحدة من IE و الضغط فقط على مفتاح F5 فى كل مرة تقوم فيها بتعديل الكود لتشاهد نتيجة التعديل .

2- **البيانات العددية**:

وهى نوعان الاعداد الصحيحة و الاعداد العشرية او ذو الدقة المضاعفة و يتم التعبير عن النوع الاول بكتابة الرقم بدون علامات تنصيص و النوع الثانى يحتوى بالطبع على كسر عشرى ولا تعطى اهتمام كبير للنوعين لان المعالج يقوم بالتحويل بين النوعين حسب نوع القيمة او ناتج العملية الحسابية لنفس المتغير وفيما يلى مثال لهذا النوع من المتغيرات :

```
$n1=3; //متغير صحيح
```

```
$n2=5.89; //متغير عشرى
$n1= $n1/$n2; // المتغيرالان اصبح عشرى
echo $n1;
```

وتكون النتيجة هى طباعة القيمة 0.509337860781

تأتى هنا اهمية معرفة العلامات الخاصة بالعمليات الحسابية وهى "+" لعملية الجمع و "-" لعملية الطرح و "/" لعملية القسمة و "*" لعملية الضرب

و هناك قاعدة هامة لاولوية الحساب فمثلا المعادلة (7+2*5) تكون نتيجتها 45 او 17 لمعرفة ذلك اتبع القاعدة التالية:
1- يتم حساب الارقام بداخل الاقواس اولا
2- يتم حساب عملية الضرب او القسمة ايهما اولا من اليسار لليمين
3- يتم حساب عملية الجمع او الطرح ايهما اولا من اليسار لليمين

مما سبق تكون المعادلة السابقة نتيجتها 17 فاذا اردت اجراء عملية الجمع اولا فيجب ان تضعها بين اقواس هكذا (7+2)*5 فتكون النتيجة 45

هذه اللغة تشبه الى حد كبير لغة C++ و يظهر ذلك عند عملية زيادة قيمة المتغير بمقدار واحد او باضافة المتغير الى نفسه مرتين كما يلى:

```
$i = $i + 1; //الوضع الاعتيادى للزيادة
$i++; //زيادة بمقدار واحد على طريقة لغة السى
$i+ = 2; //زيادة المتغير بمقدار 2
```

```
$i = $i + $i; //اضافة المتغير الى نفسه بالطريقه العادية
$i+=$i; // اضافة المتغير الى نفسه على طريقة السى
```

ولا فرق بين الطريقة العادية و طريقة السى فى النتيجة ولكن طريقة السى فقط تجعلك تبدو كمحترف اكثر

متغيرات النظام:

هى متغيرات اسمها محجوز لدى معالج اللغة بحيث يقوم باستبدال المتغير بقيمة معينة بمجرد رؤية هذا المتغير.

الثوابت:

من مكونات اى لغة برمجة وتستخدم لتخزين قيم ثابتة بحيث لا يمكن اجراء عمليات حسابية عليها وتغيير قيمتها وفائدتها للتأكد من ثبات قيمة معينة اثناء تنفيذ برنامج كبير بحيث لا يتم تغيير هذه القيمة على سبيل الخطأ .

ويتم اداء ذلك عن طريق العبارة define التى تأخذ معاملين الاول هو اسم الثابت و الثانى هو قيمة هذا الثابت و يجب ان يكون كلاهما بين علامتى التنصيص كما يلى :

```
<?
define("Pi","3.14");
echo "Pi value is... ".' '.Pi;
?>
```

وهناك ايضا ثوابت محجوزة لدى النظام مثل الثابت PHP_OS الذى يقوم بتحديد نوع نظام التشغيل المستخدم (السرفر)

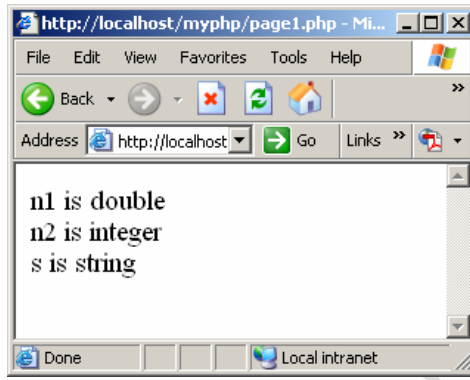
```
echo PHP_OS;
```

التحويل بين انواع البيانات:

فى بعض الاحيان يكون من الهام معرفة نوع المتغير و يتم ذلك بالنداء على دالة بسيطة تسمى gettype() و يمكن اختبارها كالتالى:

```
<?
$n1=3.5;
$n2=3;
$s="word";
echo 'n1 is '.gettype($n1).'<br>';
echo 'n2 is '.gettype($n2).'<br>';
echo 's is '.gettype($s).'<br>';
?>
```

وتكون النتيجة كالتالى :



اما اذا اردنا تحويل متغير معين من نوع الى اخر يمكن اداء ذلك عن طريق الدالة settype ويتم ذلك عن طريق معاملين الاول لاسم المتغير و الثانى لنوع المتغير الجديد كما يتضح من المثال التالى:

```
$n1= 10; //integer
settype($n1, "string");
echo gettype($n1);
```

الدالة isset():

عند كتابة برنامج كبير (اكبر من 1000 سطر) يكون من المستحيل معرفة اذا كان متغير معين قد تم استخدامه من قبل ام لا فاذا قمت مثلا بتخزين قيمة الدرجة فى متغير يسمى degree حتى تقوم فى اخر البرنامج بعرض هذه القيمة ونسيت وانت تكتب البرنامج وقمت باستخدام نفس المتغير لتخزين قيمة اخرى فستكون نتيجة الكود الذى قمت بكتابته غير صحيحه لذلك يجب اختبار المتغير اذا كنت تشك فى وجوده من قبل هكذا:

```
echo isset($n);
```

فاذا لم يكن المتغير موجود من قبل فان الدالةلا تعطى قيمه (null) واذا كان موجود فانها تعطى القيمة 1 ويمكن اختبار قيمتها عن طريق عباراة الشرط IF كما سنرى فيما بعد .

الدالة **unset()**:

وهى تحذف المتغير تماما من الذاكرة وتجعل هذا الجزء من الذاكرة صالح للاستخدام مرة اخرى يجب التأكد من عدم الحاجة الى المتغير او النداء عليه قبل استخدام هذه الدالة .

مثال:

```
unset($n);
```

الدالة **empty()**:

وهذه الدلة تختبر قيمة المتغير اذا كانت خالية " او صفر او متغير غير موجود فانها ترجع القيمة 1 و لا تقوم بارجاع اى شئ اخر خلاف ذلك .

مثال

```
$n=0;
echo pty($n).'<br>';
$n=2
echo ($n);
```

و النتيجة لن تقوم الدالة فى السطر الاول بارجاع اى قيمة وفى السطر الثانى تقوم بارجاع القيمة 1

دوال الوقت و التاريخ:

تحتوى لغة PHP على العديد من الدوال المفيدة التى تقوم باختصار الوقت لنا ويجب تذكر استخدام كل دالة حتى تستطيع استخدامها فى الوقت المناسب وفيما يلى كيفية استخدام دوال الزمن:

دالة gmdate([format]):

يتقوم هذه الدالة بارجاع قيمة الوقت و الزمن الحاليين بالتشكيل المحدد للمعامل [format] كما سنرى فأذا اردت طباعة الشهر الحالى فقط فى الصفحة فأكتب الكود :

```
<?
Echo gmdate ("m");
```

```
Echo "\t"; //مسافة خالية
Echo gmdate ("M");
?>
```

تلاحظ برنامج IE يقوم بعرض الشهر الحالى للسطر الاول كقيمة عددية ثم يقوم بطباعة مسافة خالية ثم اسم الشهر الحالى مختصر كقيمة نصية

12 Dec

من هنا نلاحظ ان استخدام الحروف الكبيرة ليست مثل استخدام الحروف الصغيرة :

مثال:
اذا اردت عرض التاريخ بالشكل يوم/شهر/سنة فقم باستخدام الدالة كالتالى:

```
echo gmdate"("d/m/Y")
```

فتكون النتيجة هى طباعة التاريخ 2004/12/30 (مثلا) فنلاحظ من هنا ان الحرف d يرمز لليوم و الحرف m يرمز للشهر و الحرف y يرمز للسنة .

أما بالنسبة للوقت فنفس القاعدة حرف h يرمز للساعة و حرف i يرمز للدقائق و حرف s يرمز للثوانى .

مثال:
الكود الاتى يقوم بطبع النتيجة: pm 12:24:53

```
echo Gmdate("h:i:s a");
```