Öğr. Gör. Serkan AKSU
http://www.serkanaksu.net

PHP'nin Temelleri

PHP Nedir?

PHP, bir programlama dili olarak, **değişkenler**, değişkenlerin değerleriyle bir işlem yapmayı sağlayan **işlemciler (operatörler)**, işlemcilerle oluşturulan deyimler ve nihayet bunların tümünü toplu olarak kullanmamızı sağlayan **işlevlere (fonksiyonlara)** sahiptir.

PHP, nesne-yönelimli (object-oriented) bir dil olduğu için, nesne oluşturma imkânına ve bunların kullanılmasını sağlayan **metodlara** da sahiptir.

Nerelerde Kullanılır?

Bütün bu imkânları kullanarak, PHP ile bir veritabanından veri alarak, bunları **HTML** etiketlerinin değerleri olarak kullanabiliriz; Web sitemizi ziyaret eden kişilerden bilgi alabiliriz, bu bilgilerle işlemler yapabiliriz. PHP'nin çeşitli komutlarını, deyimlerini ve fonksiyonlarını kullanarak, programımızın çalıştığı Web sunucusunun bulunduğu bilgisayarda çeşitli dosya işlemleri yaptırabiliriz.

Bu tür karmaşık uygulamalara geçebilmek için önce, PHP dilinin unsurlarını biraz yakından inceleyelim.

Değişkenler

Programcılıkta işlemlerimizi değişkenlerle yaparız. Değişkenler programlama dilinin kullandığı verileri tutan sembollerdir. Sözgelimi **$Gun** değişkenin adı ise bu değişkenin değeri "**Pazar**", "**Pazartesi**", "**Salı**", vb, olabilir. Her değişken, türüne göre, ya bir ya da daha fazla değer tutar.

Dizi değişkenler birden fazla değer tutabilir.

PHP'de de, birçok başka bilgisayar programlama dilinde olduğu gibi değişkenlerin içine bir değer konmadan önce tanımlanması mümkündür; fakat gerekli değildir.

Değişkenleri adının önüne $ işareti koyarak tanımlarız:

```
$degisken_adi = değeri;
```

```
$a = 5;
```

```
$isim = "Serkan";
```

Değişkenler, harf alt çizgi **(_)** ile başlayabilirler; alfanümerik karaklerleri veya rakam içerebilirler; fakat içinde boşluk, Türkçe karakler veya diğer işaretler bulunamaz. **PHP** değişkenleri her türlü değeri tutabilir: Bir değişkene ilk olarak hangi türde değer atanmışsa değişken o türde olur. Dolayısıyla, **$ad** değişkenin değeri **"Temel"** da olabilir, **1255** de olabilir.

www.serkanaksu.net

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-9" />
<title>PHP'de Giriş</title>
</head>
<body>
<?php
$ad = "Temel";
$soyad = "TAŞ";
// karakter dizileri . (nokta) ile birleştirilir.
$ad_soyad = $ad." ".$soyad;
$bolum = "Bilişim Teknolojileri";
$ortalama = 3.6;
echo "İsim : ".$ad_soyad;
echo "<br>Bölümü : ".$bolum;
echo "<br>Ortalama : ".$ortalama;
?>
</body>
</html>
```

PHP'de karakter dizilerini (stringleri) birleştirmek için **"."** (nokta) operatörü kullanılır. Yukarıdaki örnekte **$ad_soyad** değişkeninin değeri **"Temel TAŞ"** olmuştur.

Veri Türleri

PHP, değişkenlere, tuttukları değere göre farklı bellek alanı tahsis eder; bu bakımdan verilerimizin türü etkin bir programcılık açısından önem taşır. Ayrıca PHP, diğer bütün programlama dilleri gibi belirli veri türleri ile belirli işlemleri yapar veya yapamaz.

PHP'de altı tür veri tipi vardır:

Tamsayı (Integer)	**$yas** = 27;
Çift (Double)	**$pi =** 3.1415;
Alfanümerik (String)	**$isim =** "Serkan AKSU"
Mantıksal (Boolean)	**$b** = true; if(**$b**) { }

Nesne	**class SINIF**{
	var $ad;
	function ata(**$a**)
	{
	$this->**ad = **$a;
	}
	}

Dizi	**$aylar =** array ("Ocak","Şubat","Mart","Nisan","Mayıs","Haziran");

www.serkanaksu

Tür Değiştirme

Değişkenlere atadığımız değerlerinin türlerini genellikle biliriz; ama yüzlerce değişkenle uğraştığımız bir Web programında değişken türünü unutmak mümkündür. Bir değişkenin türünü öğrenmek için **gettype()** fonksiyonu kullanılır.

Şimdi, bir **PHP** programı yazalım, bir takım değişkenlere değerler atayalım ve bunların türlerini **PHP**'de tesbit edelim. Aşağıdaki kodları **turler.php** adıyla kaydedin:

turler.php

```
<?php
    $alfanumerik  = "BARTIN MYO";
    echo "Değişkenin adı: \$alfanumerik";
    echo "<br>Değeri : ".$alfanumerik;
    echo "<br>Türü : ".gettype($alfanumerik) ; //alfanümerik/string
    echo "<br><br>";

    $sayi  = 5;
    echo "Değişkenin adı: \$sayi";
    echo "<br>Değeri : ".$sayi;
    echo "<br>Türü : ".gettype($sayi) ; //tamsayi, integer
    echo "<br><br>";

    $ondalik  = 5.1234;
    echo "Değişkenin adı: \$ondalik";
    echo "<br>Değeri : ".$ondalik;
    echo "<br>Türü : ".gettype($ondalik) ; //çift,ondalık/double
    echo "<br><br>";

    $mantiksal  = true;
    echo "Değişkenin adı: \$mantiksal";
    echo "<br>Değeri : ".$mantiksal;
    echo "<br>Türü : ".gettype( $mantiksal ) ; //mantıksal/boolean
    echo "<br><br>";
?>
```

Escape

Daha önce gördük ki, tek veya çift tırnak içine de alsak, PHP, bir değişken adını gördüğü zaman onun yerine o değişkenin tuttuğu değeri yazar. PHP bunu, değişken adının başındaki $ görerek yaparak. $ işareti gibi PHP için özel anlamı olan işaretlerin anlamlandırılmasını önlemek ve bu işaretleri düz metin saymasını sağlamak için bu işaretlerin önüne ters-bölü işareti koyarız. **Buna o karakteri kurtarma veya ESCaping** denir. **PHP**'nin anlamlı işaretleri ve bunların **ESCape**-yazılışı şöyledir:

\' **Tek tırnak**

\" **Çift tırnak**

**** **Ters-bölü**

\$	**Dolar işareti**
\n	**Yeni Satır (New Line)**
\r	**Satır Başı (Return)**
\t	**Sekme (Tab) karakteri**

Değişken Kontrol Fonksiyonları

PHP'de bulunan hazır fonksiyonlardan bazıları ile değişkenlerin varlıkları ve değerleri kontrol edilebilir.

isset() : Bir fonksiyona değer atanıp atanmadığını test eder. Eğer değer atanmış ise **true** döndürür atanmamış ise **false** döndürür.

```
<?php
$degisken = 6;
if(isset($degisken)) {
    echo "<br>Değişkene değer atanmış, değeri : ".$degisken;
}
else{
    echo "<br>Değişkene değer atanmamış.";
    echo "<br>Programdan çıkılıyor";
    exit;
}
?>
```

Bu kod parçası, **$degisken** isimli değişkenin içi boş değilse, içeriğini görüntüleyecek, içi boş ise programı sonalndıracaktır.

empty() : **isset()** fonksiyonun tersi işleve sahiptir; bir değişkene değer atanmamış ise, veya değeri sıfır veya boş alfanümerik (null string) ise, **doğru (true)** değeri verir.

is_string(), is_integer(), is_double()

Sınadıkları değişkenin aradıkları türden değer içermesi halinde doğru 'True) sonuç verirler.

```
<?php
    $PI = 3.141592;
    if (is_double($PI)) {
        echo "pi Double'dır<br>";
    }else if (is_integer($PI)) {
        echo "pi Integer'dır<br>";
    }

    $i = 6;
    if (is_int($i)) {
        echo "i Integer'dır<br>";
    }

    $st = "Serkan";
    if (is_string($st)){
        echo "st String'dir<br>";
    }
?>
```

Operatörler (İşlemciler)

Bir fonksiyonun içinde de hem atama işlemi yapılabilir; hem de işlem yürütülür. PHP'nin aritmetik, birleşik-atama, birleştirme, karşılaştırma ve mantıksal sınama operatörleri vardır.

+	Toplama	6+5	=	11
-	Çıkartma	6-5	=	1
/	Bölme	6/5	=	1.2
*	Çarpma	6*5	=	30
%	Kalan (Modulus)	6%5	=	1

Aritmetik işlemleri gerçek sayılarla yaptığımız gibi, tuttuğu değer sayı olan değişkenlerle de yapabiliriz.

islem1.php

```
<?php
        $a = 6;
        $b = 22
        $c = $a + $b;
        echo  “sonuç :”.$c;
?>
```

String (Karakter Katarı) Değişkenler

Bir kişinin adı ve soyadı ayrı değişkenler halinde ise, **bunları birbirine ekleyerek (concatenation)** ortaya yeni bir **alfanümerik (sayısal olmayan)** değişken çıkartabiliriz.

String değişkenleri birleştirtmek için, **. (nokta)** operatörü kullanılır.

```
<?php
$ad = "Temel";
$soyad = "TAŞ";
// karakter dizileri . (nokta) ile birleştirilir.
$ad_soyad = $ad." ".$soyad;
$bolum = "Bilişim Teknolojileri";
$ortalama = 3.6;
echo "İsim : ".$ad_soyad;
echo "<br>Bölümü : ".$bolum;
echo "<br>Ortalama : ".$ortalama;
?>
```

Bu örnekte, sadece ekleme yoluyla yeni bir değişken oluşturmakla kalmıyoruz; fakat alfanümerik değişkenleri, başka metinlerle de birbirine ekleyebildiğimizi görüyoruz. Bu işlemi nokta işaretiyle (.) yapıyoruz. Bu işlemciyle sadece değişkenlerin değerlerini değil fakat metinleri de birbirine ekleyebiliriz:

PHP'de Atama Operatörleri

İşlemci	Örnek	Anlamı
+=	$a += 5	$a = $a + 5
-=	$a -= 5	$a = $a - 5
/=	$a /=5	$a = $a / 5
*=	$a *= 5	$a = $a * 5
%=	$a %= 5	$a = $a % 5
.=	$a .= "metin"	$a = $a." metin"

PHP'de Karşılaştırma Operatörleri

İşlemci	Örnek	Örnek	$a=6 ise:
==	eşitse	$a == 5	Yanlış/False
!=	eşit değilse	$a != 5	Doğru/True
===	aynı ise	$a === 5	Yanlış/False
>	büyükse	$a > 5	Doğuru/True
<	küçükse	$a < 5	Yanlış/False
<=	küçükse veya eşitse	$a <= 5	Yanlış/False
>=	büyükse veya eşitse	$a >= 5	Doğru/True

PHP'nin karşılaştırma işlemcileri hem tam ve ondalık sayı türü değerlerle, hem de alfanümerik değerlerle kullanılabilir.

Ve-Veya

PHP'de bu karşılaştırmayı iki grubun arasına koyduğumuz işaretlerle yaparız İşaretin sağ ve sol tarafının doğruluğu veya yanlışlığı işarete göre nihai sonucun doğru veya yanlış olmasını sağlar. Bu karşılaştırmaları yaparken şu işlemcileri kullanırız:

İşlemci	Adı	Anlamı	Örnek
||	veya	sol veya sağ doğru	doğru || yanlış = doğru
&&	ve	sol ve sağ doğru	doğru || yanlış = yanlış
Xor	Şartlı-veya	Sadece sol veya sağ doğru	doğru || yanlış = doğru
!	Değil	sol veya sağ yanlış	doğru || yanlış = doğru

"VE" (&&) ve **"VEYA" (||)** sınamaları için iki ayrı işlem işareti bulunmasının sebebi, PHP'nin işlem sırasıyla ilgilidir. Birazdan oraya geliyoruz. Şimdi, kavranması kolay olmayan ikili mantıksal sınamalara bir örnek verelim; yukarıda öğrencinin durumunu PHP komutu olarak yazalım.

www.serkanaksu.net

nothesapla.php

```php
<?php
    $vize = 45;
    $final = 65;
    $ortalama = $vize * 0.4 + $final * 0.6;
    if ($ortalama >= 60 && $final >= 50)
    {
        echo  "GEÇTİ!";
    }
    else {
        echo "KALDI!";
    }
?>
```

Öğrencinin ortalamasına göre BAŞARI NOTUNU (**$bn**) ve KATSAYISINI (**$ks**) hesaplayan PHP kodu.

```php
if($ort >= 90 && $ort <= 100) $bn = "AA";
if($ort >= 85 && $ort < 90) $bn = "BA";
if($ort >= 80 && $ort < 85) $bn = "BB";
if($ort >= 75 && $ort < 80) $bn = "CB";
if($ort >= 65 && $ort < 75) $bn = "CC";
if($ort >= 58 && $ort < 65) $bn = "DC";
if($ort >= 50 && $ort < 58) $bn = "DD";
if($ort >= 00 && $ort < 50) $bn = "FF";
switch($bn)
 {
    case "AA" : $ks = 4.0; break;
    case "BA" : $ks = 3.5; break;
    case "BB" : $ks = 3.0; break;
    case "CB" : $ks = 2.5; break;
    case "CC" : $ks = 2.0; break;
    case "DC" : $ks = 1.5; break;
    case "DD" : $ks = 1.0; break;
    case "FF" : $ks = 0.0; break;
 }
```

Bir Arttırmak veya Azaltmak

C++ da olduğu gibi değerleri sadece 1 arttırmak veya azaltmak için PHP, bir kolaylık sağlar:

$a++** veya **++$a : $a'nın değerini 1 arttırır;

$a--** veya **--$a : $a'nın değerini 1 eksiltir.

www.serkanaksu.net

Sabit Değerler

Bir programın bazen başından sonuna kadar değeri değişmeyen değişkene ihtiyacı olabilir.

Sabit değerler, değişkenlerden farklı şekilde oluşturulur. Bunun için PHP'nin define() fonksiyonunu kullanırız. Bu fonksiyonun yazım kuralı şöyledir

define ("SABIT_DEGER", değer);

Burada **SABIT_DEGER** yerine, tanımlamak istediğimiz sabit değere vereceğimiz isim, değer yerine de sabit değeri yazarız. Örnek:

sabit.php

```php
<?php
    define ( "PI", 3.14);
    $r = 4;
    $alan = PI * $r * $r;
    echo "Dairenin Alanı :".$alan;
?>
```

www.serkanaksu.net