1.Ünite - Web Programlamaya Giriş

---

1. Giriş

PHP, 1995 yılında **Rasmus Lerdorf** tarafından geliştirilen, **sunucu taraflı (server-side)** çalışan ve **HTML içine gömülebilen betik dili**dir.
Başlangıçta kişisel web sitesindeki ziyaretçi sayısını tutmak için geliştirilmiştir.
Bugün, dinamik web sayfaları oluşturmak için en popüler dillerden biridir.

- Açık kaynaklı ve ücretsizdir.
- HTML ile kolayca entegre olur.
- Sunucu tarafında çalıştığı için kullanıcı kaynak kodda PHP komutlarını göremez.
- Web sitelerinin yaklaşık %75'i PHP tabanlıdır.

---

2. PHP'nin Kısa Tarihçesi

Yıl	Olay
1993	PHP'nin ilk sürümü (Personal Home Page Tools) oluşturuldu.
1995	PHP/FI (Form Interpreter) sürümü yayınlandı.
1997	Zeev Suraski & Andi Gutmans katkısıyla PHP 3 hazırlandı, ad "PHP: Hypertext Preprocessor" oldu.
1999	Zend Motoru geliştirildi.
2000	PHP 4 (Zend Engine 1.0 ile) yayınlandı.
2004	PHP 5 – Nesneye yönelik programlama desteği.
2015	PHP 7 yayınlandı.
2020	PHP 8 sürümü çıktı.
2023	En son kararlı sürüm 8.0.9.

PHP hâlen **PHP Group** tarafından geliştirilmektedir.

---

3. Web Sunucusu ve Ortam Kurulumu

PHP'nin çalışabilmesi için bilgisayarınızda şu bileşenler bulunmalıdır:

Bileşen	Görevi
Apache	Web sunucusu (istekleri karşılar)
PHP	Betik dili yorumlayıcısı
MySQL	Veritabanı yönetim sistemi

Bu üçlü genellikle **WAMP**, **XAMPP** veya **EasyPHP** gibi hazır paketlerle birlikte kurulur.

EasyPHP Kurulumu

1. easyphp.org adresinden uygun sürüm indirilir.
2. Kurulum dizini seçilir.
3. Kurulum sonrası masaüstünde "DevServer" kısayolu oluşur.
4. Simgeye tıklayıp **HTTP Sunucusunu Başlat** seçeneğiyle sunucu çalıştırılır.
5. Kurulum test etmek için tarayıcıda şu adresler açılır:
    - http://localhost
    - http://127.0.0.1

Not: PHP dosyaları EasyPHP > www klasöründe tutulmalıdır.

---

4. Kod Editörü Kurulumu

Kod yazmak için çeşitli editörler kullanılabilir:

- **Sublime Text**
- **Notepad++**
- **PHPDesigner**
- **Dreamweaver**
- **PHPED**

Bu ders kapsamında **Sublime Text 3** önerilir:
☞ https://www.sublimetext.com/3

PHP dosyaları, EasyPHP'nin www klasörü altına kaydedilmeli ve .php uzantısı kullanılmalıdır.
Örnek:

C:\Program Files\EasyPHP\www\ilk_dosyam.php

---

5. PHP'nin Temel Özellikleri

PHP iki şekilde kullanılabilir:

1. Sunucu Taraflı Kodlama

- Dinamik içerik üretir (örneğin kullanıcı giriş sistemi, formlar, veritabanı işlemleri).
- HTML, XML, PDF veya grafik çıktıları oluşturabilir.
- MySQL, Oracle, PostgreSQL, MSSQL gibi birçok veritabanı ile çalışır.
- Apache, Nginx, IIS gibi web sunucularında çalışır.
- Windows, Linux, macOS platformlarında desteklenir.

2. Komut Satırı Uygulamaları

- PHP kodları terminal üzerinden de çalıştırılabilir.
- Sistem yönetimi, log analizi, otomasyon gibi görevlerde kullanılabilir.

---

6. PHP Kodlama Kuralları

Temel Yazım Kuralları

- PHP kodları <?php ... ?> etiketleri arasında yazılır.
- Her satır ; ile biter.
- Anahtar kelimelerde (ör. echo, while) **büyük/küçük harf duyarlılığı yoktur**.
- Değişken adlarında **duyarlılık vardır** ($ders ≠ $Ders).
- Türkçe karakter kullanılacaksa, HTML belgesine <meta charset="UTF-8"> eklenmelidir.

Örnek:

```
<meta charset="UTF-8">
<?php
  echo "Merhaba Dünya!";
  echo "İkinci komut satırı.";
?>
```

Yorum Satırları

Yorumlar, kodun açıklama bölümleridir. Tarayıcı tarafından çalıştırılmaz.

```
// Tek satırlık yorum
# Alternatif tek satırlık yorum
/*
Çok satırlı
yorumlar
buraya yazılır
*/
```

---

7. PHP – HTML Farkları

Özellik	PHP	HTML
Tür	Sunucu taraflı programlama dili	İstemci taraflı işaretleme dili
Amaç	Dinamik içerik ve arka yüz işlemleri	Sayfa yapısı ve içerik oluşturma
Dosya Uzantısı	.php	.html
Çalışma Yeri	Sunucu	Tarayıcı
Veritabanı İşlemleri	Destekler (MySQL, PostgreSQL vb.)	Desteklemez
Karmaşıklık	Orta – ileri düzey	Basit ve kolay öğrenilir
Çıktı	Dinamik (kullanıcıya göre değişir)	Statik (herkese aynı görünür)

---

8. Ek Bilgiler – isset() ve empty() Farkı

Fonksiyon	Açıklama
isset()	Değişken tanımlı mı kontrol eder.
empty()	Değişken tanımlı **ve** boş mu kontrol eder.
!isset() ≠ empty()	Birbirinin tersi değildir; farklı durumları test eder.

> Formlardan gelen veriler genellikle boş ("") döner, NULL değil.
> Bu nedenle değişkenin **tanımlı olup olmadığını** kontrol etmek için isset() kullanılır.

Sonuç

- PHP, **dinamik web geliştirme** için kullanılan açık kaynaklı bir betik dilidir.
- **Apache + PHP + MySQL** üçlüsü, web geliştirme ortamını oluşturur.
- **EasyPHP** gibi kurulum paketleri, bu bileşenleri kolayca yükler.
- PHP kodları <?php ... ?> arasında yazılır ve .php uzantısı taşır.
- **Sunucu tarafında** çalışır, kullanıcı kaynak kodda PHP'yi göremez.
- **HTML** statiktir, PHP ise **dinamik** içerik üretir.

2.Ünite - Değişken Tanımı ve Veri Türleri

PHP'de Değişken Tanımlama

PHP'de değişkenler $ işaretiyle başlar. Değişkenler herhangi bir türde veri tutabilir; tür belirtmeye gerek yoktur, PHP otomatik olarak atanan değerden türü belirler.

```
$degisken = 5;
echo $degisken; // 5
```

Kurallar:

- Değişken isimleri harf veya _ ile başlar.
- Boşluk, özel karakter ve Türkçe harf kullanılmaz.
- Harf büyüklüğüne duyarlıdır ($ad ≠ $AD).

PHP Veri Türleri

PHP'de temel olarak **altı veri türü** bulunur:

- **Integer:** Tam sayılar.

  php $yas = 25;

- **Float (Double):** Ondalıklı sayılar.

  php $oran = 1.25e3;

- **String:** Metin ifadeler.

  php $isim = "Ahmet";

- **Boolean:** TRUE veya FALSE değerleri.

  php $aktif = TRUE;

- **Array (Dizi):** Birden fazla değeri tek değişkende tutar.

  php $dersler = ["PHP", "Java", "SQL"];

- **Object (Nesne):** Sınıflardan türetilmiş veri yapıları.

  php $ogrenci = new Ogrenci();

- **NULL:** Tanımlanmamış veya boş değişken.

  php $veri = null;

Değişken Türünü Öğrenme

- **gettype()** – Değişkenin türünü döndürür.

  php echo gettype($degisken);

- **var_dump()** – Tür, değer ve uzunluğu detaylı gösterir.

  php var_dump($dizi);

Değişken Türünü Değiştirme

- **settype()** – Tür dönüşümü yapar.

  php $a = "25"; settype($a, "integer"); // Artık tamsayı

Sabitler (Constants)

Sabitler $ işareti olmadan tanımlanır ve değiştirilemez.

```
define("PI", 3.14159);
echo PI; // 3.14159
```

Kurallar:

- Harf veya _ ile başlar, büyük harfe duyarlıdır.
- Global kapsamda geçerlidir.
- defined('PI') ile tanımlı olup olmadığı kontrol edilir.

Bazı yerleşik sabitler:

__FILE__, __LINE__, __DIR__, __FUNCTION__, PHP_VERSION, PHP_OS

Değişken Kapsamı (Scope)

1. **Yerel (Local):** Fonksiyon içinde tanımlanır, sadece o fonksiyon içinde geçerlidir.

   php function test() { $a = 5; }

2. **Genel (Global):** Fonksiyon dışında tanımlanır.

   php global $a;

3. **Statik (Static):** Fonksiyon içinde tanımlanır ve çağrılar arasında değerini korur.

   php static $sayac = 0;

4. **Fonksiyon Parametreleri:** Fonksiyona gönderilen değişkenlerdir.

   php function selam($isim) { echo "Merhaba, $isim"; }

PHP Tarih ve Zaman Fonksiyonları

getdate() – Tarih ve zaman bilgisini dizi olarak döndürür.

```
$bugun = getdate();
echo "$bugun[weekday], $bugun[month] $bugun[mday], $bugun[year]";
```

Dizi anahtarları: seconds, minutes, hours, mday, wday, mon, year, weekday, month, yday

---

Sonuç

PHP'de değişkenlerin tanımı, veri türleri, tür dönüşümü, sabitler, değişken kapsamı ve tarih-zaman fonksiyonları anlatılmıştır. PHP'de veri türleri **dinamiktir**, tür kontrolü çalışma zamanında yapılır. Sabitler değiştirilemezken değişkenlerin kapsamı, tanımlandığı yere göre belirlenir.

3.Ünite - PHP Operatörler

Operatörler, değişkenler ve değerler üzerinde işlem yapmayı sağlayan sembollerdir. PHP'de başlıca altı operatör grubu bulunur:

1. Birleştirme Operatörleri
2. Aritmetik Operatörler
3. Atama Operatörleri
4. Arttırma ve Eksiltme Operatörleri
5. Karşılaştırma Operatörleri
6. Mantıksal Operatörler

---

1. Birleştirme Operatörleri

String ifadeleri birleştirmek için "." kullanılır.

```
$ad = "Ahmet";
$soyad = "Özmetin";
$adSoyad = $ad . " " . $soyad; // "Ahmet Özmetin"
```

Mevcut içeriğe ekleme yapmak için .= operatörü kullanılır:

```
$ad = "Ahmet";
$ad .= " Yıldız"; // "Ahmet Özmetin"
```

---

2. Aritmetik Operatörler

Matematiksel işlemleri gerçekleştirir:

Operatör	İşlem	Örnek
+	Toplama	$a + $b
-	Çıkarma	$a - $b
*	Çarpma	$a * $b
/	Bölme	$a / $b
%	Mod Alma	$a % $b

```
$a = 10; $b = 3;
echo $a % $b; // 1
```

---

3. Atama Operatörleri

Değişkenlere değer atamak veya mevcut değeri değiştirmek için kullanılır.

Operatör	Açıklama	Örnek
=	Atama	$a = 5;
+=	Toplayarak atama	$a += 10;
-=	Çıkartarak atama	$a -= 3;
*=	Çarparak atama	$a *= 2;
/=	Bölerek atama	$a /= 4;
%=	Mod alarak atama	$a %= 2;
.=	Birleştirerek atama	$metin .= " eklendi";

---

4. Arttırma ve Eksiltme Operatörleri

Değişkenin değerini 1 artırır veya azaltır.

- ++$x: Önceden arttırma
- $x++: Sonradan arttırma
- --$x: Önceden azaltma
- $x--: Sonradan azaltma

```
$x = 5;
echo ++$x; // 6
echo $x--; // 6, ardından 5
```

Not: PHP'de karakterler üzerinde de kullanılabilir:

```
$harf = 'a';
$harf++; // 'b'
$harf = 'z';
$harf++; // 'a'
```

---

5. Karşılaştırma Operatörleri

İki değeri karşılaştırır, sonuç TRUE veya FALSE döner.

Operatör	Anlamı	Örnek
==	Eşittir	$a == $b
===	Tür ve değer olarak denktir	$a === $b
!= veya <>	Eşit değildir	$a != $b
!==	Denk değildir	$a !== $b
<	Küçüktür	$a < $b
>	Büyüktür	$a > $b

Operatör	Anlamı	Örnek
<=	Küçük eşit	$a <= $b
>=	Büyük eşit	$a >= $b

---

6. Mantıksal Operatörler

Mantıksal ifadeleri birleştirir.

Operatör	Anlamı	Açıklama
and veya &&	VE	Her iki ifade doğruysa TRUE
or veya ||	VEYA	Sadece biri doğruysa TRUE
xor	Ayrıcalıklı VEYA	Sadece biri doğruysa TRUE
!	DEĞİL	İfade yanlışsa TRUE

İşlem Önceliği (Operator Precedence)

PHP, işlemleri matematiksel öncelik sırasına göre yürütür:

1. () Parantez
2. ++, --
3. *, /, %
4. +, -
5. <, <=, >, >=
6. ==, ===, !=, !==
7. &&
8. ||
9. =, +=, -=, *=, /=
10. AND
11. XOR
12. OR

Sonuç

Bu ünitede PHP'de kullanılan operatörler detaylı olarak incelenmiştir.
Operatörler; matematiksel, mantıksal, karşılaştırmalı veya birleştirici işlemler yapmayı sağlar.
İşlem önceliği, parantez → arttırma/azaltma → çarpma/bölme → toplama/çıkarma → karşılaştırma → mantıksal işlemler şeklindedir.
Operatörlerin doğru kullanımı, PHP'de koşullar ve hesaplamalar için temel öneme sahiptir.

4.Ünite - Diziler

Diziler, birden fazla veriyi tek bir değişken içinde saklamayı sağlar. Her eleman bir "indis" (index) veya "anahtar" (key) ile tanımlanır. PHP'de dizi oluşturmak için iki sözdizimi vardır:

```
$dersler = array("PHP", "Java", "SQL");
$dersler = ["PHP", "Java", "SQL"];
```

Diziler farklı türdeki verileri (sayı, metin, boolean) aynı anda tutabilir.

Dizilerin Görüntülenmesi

`echo` yalnızca bir dizi olduğunu belirtir. Dizinin içeriğini görüntülemek için:

```
print_r($dizi);
var_dump($dizi);
```

Bu fonksiyonlar, dizinin elemanlarını indisleriyle birlikte listeler.

İndisli (Numaralı) Diziler

Sıfırdan başlayan sayısal indekslere sahip dizilerdir.

```
$isim = array("Ayşe", "Ali", "Ahmet");
echo $isim[0]; // Ayşe
```

Döngüyle tüm elemanlara ulaşmak için:

```
$dersler = array("Web", "Java", "PHP");
for($i = 0; $i < count($dersler); $i++) {
  echo $dersler[$i];
}
```

İlişkisel Diziler

Her elemanın anahtar–değer şeklinde saklandığı dizilerdir. Anahtar string veya sayı olabilir.

```
$derskredi = array("WEB" => 6, "JAVA" => 4, "PHP" => 5);
echo $derskredi["WEB"]; // 6
```

Aynı işlem `=>` operatörüyle manuel olarak da yapılabilir:

```
$derskredi["WEB"] = 6;
```

Tırnaksız anahtar kullanımı uyarıya yol açabilir, bu nedenle anahtarlar tırnak içinde yazılmalıdır.

Çok Boyutlu Diziler

Bir dizi içinde başka dizilerin bulunduğu yapılardır.

```
$dersler = [
  "PHP" => ["Eğitmen" => "Ali", "Hafta" => "12", "Kredi" => 5],
  "WEB" => ["Eğitmen" => "Ahmet", "Hafta" => "15", "Kredi" => 4]
];
echo $dersler["PHP"]["Eğitmen"]; // Ali
```

Bu tür diziler karmaşık veri yapılarının düzenli tutulmasını sağlar.

Dizi Fonksiyonları

- **count()**: Dizi eleman sayısını döndürür.

  php echo count($dersler); // 3

- **is_array()**: Değişkenin dizi olup olmadığını kontrol eder.

  php echo is_array($dersler); // 1 (true)

- **implode()**: Dizi elemanlarını birleştirip string döndürür.

  php echo implode(',', $dersler); // PHP,Java,SQL

- **explode()**: String ifadeyi ayraçlara göre parçalayarak diziye çevirir.

  php $dersler = explode(',', "PHP,Java,SQL");

Sıralama Fonksiyonları

Fonksiyon	Açıklama

Fonksiyon	Açıklama
sort()	Değere göre küçükten büyüğe sıralar, anahtarları bozar.
rsort()	Değere göre büyükten küçüğe sıralar, anahtarları bozar.
asort()	Değere göre küçükten büyüğe sıralar, anahtarları korur.
arsort()	Değere göre büyükten küçüğe sıralar, anahtarları korur.
ksort()	Anahtara göre küçükten büyüğe sıralar.
krsort()	Anahtara göre büyükten küçüğe sıralar.

```
$dersler = array("PHP", "WEB", "JAVA");
sort($dersler);
print_r($dersler); // JAVA, PHP, WEB
```

Sonuç

- PHP dizileri farklı türde verileri bir arada saklayabilir.
- Diziler; **indisli**, **ilişkisel** ve **çok boyutlu** olarak üçe ayrılır.
- print_r(), var_dump(), count(), implode(), explode() gibi fonksiyonlar dizilerle çalışmayı kolaylaştırır.
- Sıralama fonksiyonları ile diziler değer veya anahtara göre düzenlenebilir.

5.Ünite - Koşul İfadeleri

Programlamada **karar verme** işlemleri koşul ifadeleriyle yapılır.
Koşullar, belirli bir mantıksal ifadeye göre farklı işlemler yürütülmesini sağlar.
PHP'de koşul ifadeleri genellikle **if, else, elseif, switch** ve **üçlü (ternary)** yapılarla oluşturulur.

---

1. if Yapısı

Bir koşul doğruysa (TRUE) kod bloğu çalışır.

```
if ($yas >= 18) {
  echo "Reşitsiniz.";
}
```

---

2. if – else Yapısı

Koşul sağlanmazsa (FALSE) alternatif kod bloğu yürütülür.

```
if ($yas >= 18) {
  echo "Reşitsiniz.";
} else {
  echo "Reşit değilsiniz.";
}
```

---

3. if – elseif – else Yapısı

Birden fazla koşulun sırayla kontrol edilmesini sağlar.

```
if ($puan >= 90) {
  echo "AA";
} elseif ($puan >= 80) {
  echo "BA";
} else {
  echo "Kaldınız";
}
```

---

4. İç İçe (Nested) if Yapıları

Bir if bloğunun içinde başka bir if yer alabilir.

```
if ($giris == true) {
  if ($rol == "admin") {
    echo "Yönetici paneline hoş geldiniz.";
  }
}
```

---

5. Ternary (Üçlü) Operatör

Kısa if–else yazımıdır:

```
$sonuc = ($yas >= 18) ? "Reşit" : "Reşit değil";
```

Bu yapı, basit koşullarda kodu daha okunabilir hale getirir.

---

6. Switch – Case Yapısı

Bir değişkenin alabileceği farklı değerlere göre işlem yapılmasını sağlar.

```
switch ($gun) {
  case "Pazartesi":
    echo "Hafta başı";
    break;
  case "Cuma":
    echo "Hafta sonu yaklaşıyor";
    break;
  default:
    echo "Hafta içi günlerinden biri";
}
```

break komutu kullanılmazsa sonraki durumlar da çalışır (“fall-through” davranışı).

---

7. Match İfadesi (PHP 8+)

switch yapısına benzer ama **katı karşılaştırma** (===) yapar ve **değer döndürür**.

```
$sonuc = match($puan) {
  100 => "Mükemmel",
  90, 80 => "Başarılı",
  default => "Orta"
};
```

---

8. Mantıksal Operatörlerle Koşullar

Koşullar AND, OR, ! operatörleriyle birleştirilebilir.

```
if ($yas >= 18 && $uyelik == true) {
  echo "Giriş başarılı.";
}
```

Sonuç

Bu bölümde PHP’de karar yapıları tanıtılmıştır.

- **if** ve **else** koşul temelli akış sağlar.
- **switch** çoklu seçeneklerde daha düzenli yapı sunar.
- **match** ifadesi, modern PHP’de güvenli ve kısa bir alternatiftir.
- Mantıksal operatörler, birden fazla koşulun birlikte değerlendirilmesini sağlar.

6.Ünite - Döngüler

PHP programlama dilinde belirli kod bloklarının tekrarlanmasını sağlayan**döngü (loop)** yapılarını, söz dizimlerini ve kullanım senaryolarını ele almaktadır.

1. PHP'de Kullanılan Temel Döngüler

PHP'de dört temel döngü türü bulunmaktadır:

- **For Döngüsü:** Döngünün kaç kez çalışacağının önceden bilindiği durumlarda kullanılır.
    - En güçlü döngü türü olarak kabul edilir; başlangıç değeri, koşul ve artış miktarı tek satırda tanımlanır.
    - **Yapısı:** for (sayaç; koşul; değişim) { kodlar }.

```
<?php
// Başlangıç 1, Bitiş 10, Artış +1
for ($i = 1; $i <= 10; $i++) {
   echo $i;      // Değeri yazdır
   echo "<br>";
}
?>
```

- **While Döngüsü:** Belirtilen koşul sağlandığı (doğru olduğu) sürece çalışmaya devam eder.
    - Koşul, döngüye girmeden **önce** kontrol edilir. Eğer koşul baştan sağlanmıyorsa döngü hiç çalışmaz.

```
<?php
$toplam = 0;
$sayi = 0;
while ($sayi < 100) {
   $toplam += $sayi; // Sayıyı toplama ekle
   $sayi++;          // Sayıyı 1 artır
}
echo "Sayıların Toplamı =" . $toplam;
?>
```

- **Do-While Döngüsü:** while döngüsüne benzer ancak koşul kontrolü döngü **sonunda** yapılır.
    - Bu yapı sayesinde, koşul sağlanmasa bile döngü içindeki kodlar **en az bir kez** mutlaka çalıştırılır.

```
<?php
$i = 0;
do {
   echo $i;
} while ($i > 0);
?>
```

- **Foreach Döngüsü:** Sadece **diziler (arrays)** ve nesneler üzerinde işlem yapmak için kullanılır.
    - Dizideki eleman sayısı kadar çalışır.
    - Hem sadece değer (as $deger) hem de anahtar-değer (as $anahtar => $deger) ikilisiyle kullanılabilir.

```
<?php
$dizi = [
   "1" => "Javascript",
   "2" => "PHP",
   "3" => "Bootstrap",
   "4" => "JQuery"
];

foreach ($dizi as $key => $deger) {
   echo "Anahtar $key => Değer $deger" . "<br/>";
}
?>
```

2. Döngü Kontrol İfadeleri

Döngülerin akışını değiştirmek veya sonlandırmak için özel deyimler kullanılır:

- **Break Deyimi:** Döngüyü tamamen sonlandırmak için kullanılır.
    - İç içe döngülerde, break yanına sayı yazılarak (örn: break 2;) kaç katman dışarı çıkılacağı belirtilebilir.
- **Continue Deyimi:** Döngünün o anki adımını (yinelemesini) atlayıp, kodları çalıştırmadan bir sonraki adıma geçmesini sağlar.
    - Hata oluşumunu önlemek veya gereksiz işlem yapmamak için kullanılır (örn: sıfıra bölme hatasını engellemek).
- **Goto Deyimi:** Program akışının kod içinde belirlenen etiketli (label) başka bir noktaya atlamasını sağlar.

- Döngü veya switch yapılarının içine dışarıdan atlanamaz, ancak içinden dışarıya çıkılabilir.

3. Önemli Karşılaştırmalar ve İpuçları

- **While vs Do-While:** while koşulu başta kontrol eder (0 kez çalışabilir), do-while sonda kontrol eder (en az 1 kez çalışır).
- **Sonsuz Döngü:** Döngü koşulu sürekli true dönerse veya sayaç değişkeni değiştirilmezse döngü sonsuza kadar çalışır.
- **İç İçe Döngüler:** Bir döngü bloğu içerisinde başka bir döngü kullanılabilir (örn: çok boyutlu dizileri gezmek için).

7.Ünite - Fonksiyonlar

PHP'de kod tekrarını önlemek, yönetimi kolaylaştırmak ve performansı artırmak için kullanılan **fonksiyon** yapılarını, türlerini ve kullanım şekillerini ele almaktadır.

1. Fonksiyonların Temel Yapısı ve Avantajları

Fonksiyon, belirli bir görevi yerine getiren ve istendiğinde çağrılabilen kod bloğudur.

- **Avantajları:** Daha az satır kod, hata oranını düşürme, daha hızlı yükleme ve çalışma süresi sağlar.
- **Söz Dizimi:** function isim(parametreler) { komutlar; return değer; } şeklindedir. İsimler büyük/küçük harfe duyarlı değildir.

2. Parametreler ve Değer Döndürme

Fonksiyonlar dışarıdan veri alabilir ve işledikleri sonucu dışarı aktarabilirler.

- **Parametreli/Parametresiz:** Fonksiyonlar boş çalışabileceği gibi parantez içine değişkenler (argümanlar) alarak da çalışabilir.
- **Varsayılan Değer:** Parametreye varsayılan bir değer atanabilir (örn: $a=0). Fonksiyon çağrılırken değer girilmezse bu varsayılan değer kullanılır.
- **Return Deyimi:** Fonksiyon içinde üretilen değerin, fonksiyonun çağrıldığı yere geri gönderilmesini sağlar.

3. Değişken Kapsamı ve Erişim Türleri

PHP'de değişkenlerin nerede geçerli olduğu (scope) önemlidir.

- **Yerel ve Global Değişkenler:** Fonksiyon içinde tanımlanan değişkenler **yereldir** ve dışarıdan erişilemez. Dışarıdaki bir değişkeni fonksiyon içinde kullanmak için global anahtar kelimesi kullanılır.
- **Referans ile Gönderme (&):** Normalde değişkenin kopyası gönderilir. Ancak değişkenin başına & konulursa, fonksiyon içindeki değişiklikler ana değişkeni de etkiler (orijinal veriyi değiştirir).

4. Özel Fonksiyon Türleri

- **İç İçe Fonksiyonlar:** Bir fonksiyonun içinde başka bir fonksiyon tanımlanabilir. İçteki fonksiyon, dıştaki çağrılmadan erişilemez.
- **Recursive (Özyinelemeli) Fonksiyonlar:** Kendi kendini çağıran fonksiyonlardır. Genellikle karmaşık matematiksel hesaplamalar veya döngüsel işlemler için kullanılır.

5. Hazır (Yerleşik) Fonksiyonlar

PHP'de geliştiricinin tanımladıklarının (User Defined) yanı sıra dilin sunduğu hazır fonksiyonlar (Built-in) vardır:

- **Metinsel:** strlen (uzunluk), trim (boşluk temizleme), strpos (metin içinde arama), ltrim/rtrim (sağ/sol boşluk silme).
- **Matematiksel:** rand (rastgele sayı), sqrt (karekök), max/min (en büyük/küçük), round (yuvarlama).
- **Tarih-Zaman:** date (tarih biçimlendirme), time (Unix zaman damgası).

Kod Örnekleri

1. Varsayılan Parametreli Fonksiyon

Değer gönderilmezse varsayılanı kullanır.

```
<?php
function cevreHesapla($kisa, $uzun = 0) {
    if ($uzun == 0) {
        return $kisa * 4; // Kare çevresi
    } else {
        return 2 * ($kisa + $uzun); // Dikdörtgen çevresi
    }
}
echo cevreHesapla(5); // Çıktı: 20 (Kare)
echo cevreHesapla(5, 10); // Çıktı: 30 (Dikdörtgen)
?>
```

2. Global Değişken Kullanımı

Fonksiyon dışındaki değişkene erişim sağlar.

```
<?php
$x = 10;
function topla($y) {
    global $x; // Dışarıdaki $x'i içeri al
    return $x + $y;
}
echo topla(5); // Çıktı: 15
?>
```

3. Referans ile Değer Değiştirme

Değişkenin orijinal değerini değiştirir.

```
<?php
$sayi = 10;
function arttir(&$deger) { // & işareti referans olduğunu belirtir
    $deger += 5;
}
arttir($sayi);
echo $sayi; // Çıktı: 15 (Orijinal $sayi değişti)
?>
```

4. Recursive (Kendini Çağıran) Fonksiyon

Belli bir şarta kadar kendini tekrar eder.

```
<?php
$deneme = 1;
function tahmin($hedef) {
    global $deneme;
    if ($hedef != rand(1, 3)) { // Rastgele sayı tutmazsa
        $deneme++;
        tahmin($hedef); // Fonksiyon kendini tekrar çağırır
    } else {
        echo "$deneme. denemede bulundu.";
    }
}
tahmin(2);
?>
```

8.Ünite - PHP Formlar

Kullanıcıların web siteleriyle etkileşime girmesini sağlayan HTML formlarının oluşturulmasını, bu formlardan gelen verilerin PHP ile nasıl yakalanacağını (GET ve POST metodları) ve sunucu tarafında nasıl doğrulanacağını ele almaktadır.

1. HTML Form Yapısı

Web formları, kullanıcıdan veri almak için kullanılır ve `<form>` etiketleri arasında tanımlanır. Bir formun üç temel özelliği vardır:

- **Action:** Form verilerinin gönderileceği hedef sayfayı belirtir (örneğin:`islem.php`).
- **Method:** Verilerin gönderilme şeklini (GET veya POST) belirler.

- **Form Elemanları:** Veri girişi sağlayan araçlardır.

Temel Form Elemanları:

- **Input:** En çok kullanılan etikettir. `type` özelliği ile şekillenir (text, password, radio, checkbox, file, submit).
- **Textarea:** Çok satırlı geniş metin girişleri için kullanılır.
- **Select** / **Option:** Açılır liste menüleri oluşturur.
- **Fieldset** / **Legend:** Form elemanlarını gruplamak ve başlık eklemek için kullanılır.

2. Form Gönderme Metotları (GET ve POST)

Form verileri sunucuya iki ana yöntemle gönderilir:

POST Metodu

- Veriler URL'de **görünmez**, arka planda gönderilir.
- Güvenlik gerektiren veriler (şifre vb.) için uygundur.
- Veri boyutunda bir kısıtlama yoktur.
- PHP'de verilere `$_POST['name_degeri']` dizisi ile ulaşılır.

GET Metodu

- Veriler URL adres çubuğunda görünür (örn: `index.php?ad=ahmet&yas=30`).
- Güvenli değildir, hassas veriler için kullanılmamalıdır.
- Veri boyutu sınırlıdır (maksimum 2048 karakter).
- Sayfalar yer imlerine eklenebilir.
- PHP'de verilere `$_GET['name_degeri']` dizisi ile ulaşılır.

3. Form Doğrulama (Validation)

Formdan gelen verilerin işlenmeden önce kontrol edilmesi gerekir. Bunun için iki temel fonksiyon kullanılır:

- **isset():** Bir değişkenin tanımlı olup olmadığını kontrol eder. Genellikle formun gönderilip gönderilmediğini anlamak için kullanılır (örn: Gönder butonuna basıldı mı?).
- **empty():** Bir değişkenin boş olup olmadığını kontrol eder. `0`, `""` (boş string), `NULL` veya `FALSE` değerleri boş kabul edilir.

Kod Örnekleri

1. Basit HTML Form Örneği

Kullanıcıdan ad ve şifre alan bir yapı.

```
<form action="kontrol.php" method="post">
   Kullanıcı Adı: <input type="text" name="kullanici_adi"><br>
   Şifre: <input type="password" name="sifre"><br>
   <input type="submit" value="Giriş Yap">
</form>
```

2. PHP ile Verileri Yakalama (POST)

Formdan gelen verileri `$_POST` süper global dizisi ile alma.

```
<?php
// Form gönderildiğinde verileri ekrana yazar
echo "Girilen Kullanıcı: " . $_POST["kullanici_adi"];
echo "Girilen Şifre: " . $_POST["sifre"];
?>
```

3. Form Doğrulama (Isset ve Empty)

Veri gönderilmiş mi ve içi dolu mu kontrolü.

```
<?php
// 1. Kontrol: Form gönderilmiş mi? (Buton tanımlı mı?)
if (isset($_POST['kullanici_adi']) && isset($_POST['sifre'])) {

    $kadi = $_POST['kullanici_adi'];
    $sifre = $_POST['sifre'];

    // 2. Kontrol: Alanlar boş mu?
    if (empty($kadi) || empty($sifre)) {
        echo "Lütfen tüm alanları doldurunuz!";
    } else {
        echo "Giriş Başarılı: " . $kadi;
    }

} else {
    echo "Lütfen formu kullanarak giriş yapınız.";
}
?>
```

9.Ünite - Cookies (Çerezler)

Web sitelerinin kullanıcıyı hatırlamasını sağlayan **çerez (cookie)** yapısını, PHP ile çerez oluşturma, okuma ve silme işlemlerini ve bu teknolojinin avantaj/dezavantajlarını ele almaktadır.

1. Çerez (Cookie) Nedir?

Çerez, bir web sunucusu tarafından tarayıcı (browser) aracılığıyla kullanıcının bilgisayarına kaydedilen küçük veri parçacığıdır.

- **Amacı:** Kullanıcıyı hatırlamak (beni hatırla), oturum bilgilerini saklamak, alışveriş sepetini korumak ve kişiselleştirilmiş içerik sunmaktır.
- **Erişim Kuralı:** Bir çerez, yalnızca onu oluşturan alan adı (domain) tarafından okunabilir. Başka siteler bu verilere erişemez.
- **Çalışma Mantığı:** HTML kodları gönderilmeden önce, başlık (header) transferi sırasında sunucu ile tarayıcı arasında iletilir. Bu yüzden kod yazarken **HTML çıktısından önce** tanımlanmalıdır.

2. PHP'de Çerez Yönetimi

PHP'de çerez işlemleri `setcookie()` fonksiyonu ve `$_COOKIE` süper global dizisi ile yönetilir.

Çerez Oluşturma (`setcookie`)

Söz dizimi: `setcookie(isim, değer, süre, yol, alan_adı, güvenli, http_only)`

- **İsim:** Çerezin çağrılacağı addır (Zorunlu).
- **Değer:** İçerisinde saklanacak veridir.
- **Süre:** Saniye cinsinden geçerlilik süresidir (Unix zaman damgası). Belirtilmezse tarayıcı kapanınca silinir.
- **Yol:** Çerezin sitenin hangi dizinlerinde geçerli olacağını belirtir (Genelde `/` kullanılarak tüm sitede aktif edilir).

Çerez Okuma ve Kontrol Etme

- Tarayıcıda kayıtlı çerezlere `$_COOKIE['cerez_adi']` dizisi ile ulaşılır.
- Çerezin varlığını kontrol etmek için `isset()` fonksiyonu kullanılır.

Çerez Silme ve Değiştirme

- **Değiştirme:** Aynı isimle yeni bir `setcookie()` tanımlaması yapılarak değeri güncellenir.
- **Silme:** Çerezi silmek için `setcookie()` fonksiyonu, **geçmiş bir zaman dilimi** (örneğin 1 saat öncesi) verilerek tekrar çalıştırılır.

3. Avantajlar ve Dezavantajlar

- **Avantajlar:** Kullanıcı deneyimini kişiselleştirir, oturumların açık kalmasını sağlar, e-ticaret sitelerinde sepet takibi yapar ve site analitiği için veri sağlar.
- **Dezavantajlar:** Gizlilik endişeleri yaratabilir (kullanıcı izleme), güvenlik riskleri oluşturabilir (hassas veriler saklanmamalıdır) ve kullanıcılar tarayıcı ayarlarından çerezleri engelleyebilir.

Kod Örnekleri

Konuyu pekiştirmek için ders notlarından türetilmiş temel örnekler:

1. Çerez Oluşturma (Süreli)

Bir kullanıcı adını 1 saatliğine (3600 saniye) kaydetme.

```
<?php
// Şimdiki zaman + 3600 saniye
setcookie("kullanici_adi", "Istanbul Universitesi", time() + 3600);
?>
```

2. Çerez Okuma ve Kontrol Etme

Çerez varsa ekrana yazdıran, yoksa uyarı veren yapı.

```
<?php
if (isset($_COOKIE["kullanici_adi"])) {
    echo "Hoşgeldin: " . $_COOKIE["kullanici_adi"];
} else {
    echo "Çerez bulunamadı veya süresi doldu.";
}
?>
```

3. Çerez Silme

Çerezin süresini geçmişe ayarlayarak tarayıcıdan silinmesini sağlama.

```
<?php
// Zamanı geçmişe (örneğin 1 saat geriye) alarak silme işlemi
setcookie("kullanici_adi", "", time() - 3600);
?>
```

10.Ünite - Session (Oturum Yönetimi)

Web uygulamalarında kullanıcı verilerini sunucu tarafında geçici olarak saklamayı sağlayan **Session (Oturum)** yapısını, çerezlerden farkını ve temel kullanım komutlarını ele almaktadır.

1. Session (Oturum) Nedir?

Session, kullanıcıların web sitesindeki etkinlikleri süresince verilerin **sunucu tarafında** (server-side) saklanmasını sağlayan bir mekanizmadır.

- **Kullanım Alanları:** Kullanıcı girişi (login), alışveriş sepeti yönetimi ve sayfa geçişlerinde veri taşıma.
- **Çalışma Mantığı:** Kullanıcı siteye girdiğinde sunucu ona benzersiz bir kimlik (**PHPSESSID**) verir. Bu kimlik genellikle kullanıcının tarayıcısında bir çerez olarak tutulur, ancak asıl veriler sunucuda saklanır.
- **Analoji:** Session mantığı kredi kartına benzer. Kart (Session ID) sizdedir, ancak para ve hesap bilgileri (Veri) bankada (Sunucuda) durur.

2. Session Yönetimi ve Komutlar

PHP'de oturumlar `$_SESSION` süper global dizisi ile yönetilir.

- **Başlatma (`session_start()`):** Session kullanımı yapılacak her sayfanın **en başında** mutlaka çağrılmalıdır. Aksi takdirde oturum verilerine erişilemez.

- **Veri Atama:** $_SESSION dizisine anahtar-değer ilişkisi ile veri eklenir. Dizi (array) türünde veriler de saklanabilir.
- **Veri Silme (unset()):** Sadece belirli bir oturum değişkenini silmek için kullanılır (Örn: Sadece sepeti boşaltmak).
- **Oturumu Yok Etme (session_destroy()):** Mevcut oturumu tamamen kapatır ve tüm verileri siler (Örn: Güvenli çıkış yapma).

3. Session ve Cookie Arasındaki Farklar

Session ve Cookie, veri saklama yöntemleri olsa da temel farkları şunlardır:

- **Depolama Yeri:** Session verileri **sunucuda**, Cookieler **kullanıcı bilgisayarında** (tarayıcıda) saklanır.
- **Güvenlik:** Session, veriler sunucuda kaldığı için Cookielere göre daha güvenlidir.
- **Süre:** Session genellikle tarayıcı kapatıldığında silinir. Cookieler ise belirlenen son kullanma tarihine kadar saklanabilir.
- **Kapasite:** Session depolama alanı sunucu kapasitesiyle sınırlıdır (teorik olarak sınırsız), Cookieler ise 4KB ile sınırlıdır.

Kod Örnekleri

1. Session Başlatma ve Veri Oluşturma

Oturumu başlatıp kullanıcı adını kaydetme.

```
<?php
session_start(); // Session kullanımı için zorunludur

// 'mesaj' adında bir session oluşturup değer atama
$_SESSION['mesaj'] = "Merhaba Dünya";

// Dizi (Array) olarak veri saklama
$_SESSION['uyeler'] = array("Ahmet", "Mehmet", "Ayşe");
?>
```

2. Başka Bir Sayfada Veriye Erişme

Farklı bir sayfada, önceden oluşturulmuş veriyi okuma.

```
<?php
session_start(); // Bu sayfada da başlatılmalı

if (isset($_SESSION['mesaj'])) {
    echo "Kaydedilen Mesaj: " . $_SESSION['mesaj'];
} else {
    echo "Oturum verisi bulunamadı.";
}
?>
```

3. Oturumu Sonlandırma (Çıkış İşlemi)

Tüm oturum verilerini temizleyip oturumu kapatma.

```
<?php
session_start();

// 1. Tüm değişkenleri temizle
session_unset();

// 2. Oturumu tamamen yok et
session_destroy();

echo "Başarıyla çıkış yapıldı.";
?>
```