

# HTML / CSS – DUYARLI(RESPONSİVE) TASARIM

- HTML ve CSS Temelleri
- Duyarlı (Responsive) Tasarım



# HTML – Hyper Text Markup Language

(Yüksek Metin İşaretleme Dili)

---

- HTML, web sayfalarını oluşturmak için kullanılan standart [metin işaretleme dilidir](#).
- Web sitelerinin şekillendirilmesinde, içerik düzenleme ve yapılandırma kullanılır.
- HTML dosyasının uzantısı **\*.htm** veya **\*.html** olmak zorundadır (farklı amaçlar ile framework veya uygulamaya özel dosya tipi kullanılmadıkça).



# HTML Sayfası Temel Yapısı

```
<!DOCTYPE Html>  
<html>  
  <head>  
    <title> Benim Sayfam </title>  
    <meta charset="UTF-8">  
  </head>  
  <body>  
    <h1>Sayfama Hoş geldiniz!</h1>  
  </body>  
</html>
```

# HTML Etiketleri

- **Başlık etiketleri:** `<h1>` ve `<h6>` dâhil olmak üzere aradaki tüm rakamlar kullanılarak tanımlanabilir
- **Paragraf etiketi:** `<p> Bu bir paragraf </p>`
- **Satır etiketi :** Satır atlama etiketi `<br>` veya `<br/>` etiketidir, bu etiketin bitiş etiketi yoktur.
- **Yorum etiketi:** `<!-- Bu bir açıklama -- >`
- **HTML Links – Hyperlinks:** Html sayfalarında bir başka sayfaya, içeriğe ya da url adresine bağlantı oluşturabiliriz
  - **href:** Hangi bağlantıya gideceğini burada belirtiriz.
  - **target:** Sayfanın nerede açılacağını burada belirtiriz. Örneğin `target="_blank"` Bağlantıyı yeni bir pencerede açar.
    - `<a href="images.html" > FOTOĞRAFLAR </a>` yazıya tıklayınca **images.html** sayfası açılacak:
    - `<a href="https://www.google.com" > Google </a>` yazıya tıklayınca ilgili web adresi açılacak.
- **Resimler:** ``

# HTML Etiketleri

## LİSTELER:

- **Sırasız Listeler:** Sırasız bir liste maddelerden oluşur. `<ul>`  
`<li>...</li><li>...</li></ul>`
- **Sıralı Listeler:** Maddeler rakamlar, roma rakamları veya harfler ile listelenir. `<ol>`  
`<li>...</li><li>...</li></ol>`

```
<h3> Alışveriş Listesi: </h3>
```

```
<ul>
  <li>Kahve</li>
  <li>Süt</li>
  <li>Çikolata</li>
</ul>
```

Alışveriş Listesi:

- Kahve
- Süt
- Çikolata

```
<h3> Alışveriş Listesi: </h3>
```

```
<ol>
  <li>Kahve</li>
  <li>Süt</li>
  <li>Çikolata</li>
</ol>
```

Alışveriş Listesi:

1. Kahve
2. Süt
3. Çikolata

```
<ol type="A">
  <li>Sucuk</li>
  <li>Menemen</li>
  <li>Bal</li>
</ol>
```

Alışveriş Listesi:

- A. Sucuk
- B. Menemen
- C. Bal

# HTML Etiketleri

## TABLolar:

- **<table>** Tablonun başlangıç ve bitiş etiketidir.
- **<tr>** Tablo satırı ekleme etiketidir.
- **<td>** Tablo veri hücresi ekleme etiketidir. Sütun ekleme olarak da düşünebiliriz.
- **<th>** Tablo başlığı ekleme etiketidir.
- **<thead>** Tablo başlığını belirgin hale getirmek için kullanılır. Görünüşe etkisi yoktur.
- **<tbody>** Tablo gövdesini belirgin hale getirmek için kullanılır. Görünüşe etkisi yoktur.

```
<table border="2">
  <thead>
    <tr>
      <th>İsim</th>
      <th>Ödev</th>
      <th>Proje</th>
      <th>Vize</th>
      <th>Final</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Volkan</td>
      <td> 50 </td>
      <td> 90 </td>
      <td> 80 </td>
      <td> 100 </td>
    </tr>
  </tbody>
</table>
```

İsim	Ödev	Proje	Vize	Final
Volkan	50	90	80	100

# HTML Etiketleri

## FORMLAR:

- **input:** Kullanıcının bilgi girişi yapması gereken yerlerde kullanılır.
- **type:** Formun türünü belirler.
- **text:** Metin girişi yapılması içindir.
- **name:** Değişken değerini belirler. Her input değeri için farklı isim vermemiz zorunludur.
- **value:** input elemanı içerisine varsayılan olarak yazı yazmamızı sağlar

## Bazı input ve textarea parametreleri:

- **autofocus:** İlgili metin alanını seçili hale getirir ve böylece görsel olarak öne çıkmasını sağlar.
- **disabled:** Veri girişini pasif hale getirir.
- **readonly:** Sadece okunabilir metin gözükür. Bu alanı fare ile seçemeyiz.
- **required:** Doldurulması zorunlu alanlarda bu parametreyi kullanırız.

<input type="button">	<input type="button" value="Button"/>
<input type="checkbox">	<input type="checkbox"/>
<input type="color">	<input type="color"/>
<input type="date">	<input type="date"/>
<input type="email">	<input type="email" value="farazc60@gmail.com"/>
<input type="file">	<input type="file"/>
<input type="hidden">	<input type="hidden"/>
<input type="image">	<input type="image"/>
<input type="number">	<input type="number" value="5"/>
<input type="password">	<input type="password"/>
<input type="radio">	<input type="radio"/>
<input type="range">	<input type="range"/>
<input type="reset">	<input type="reset" value="Reset"/>
<input type="submit">	<input type="submit" value="Submit"/>
<input type="text">	<input type="text" value="codewithfaraz"/>
<input type="url">	<input type="url" value="https://www.codewithfaraz.com"/>

# HTML Etiketleri

## input type parametresine ait diğer özellikler:

- **password:** Şifreli alanlarda bu özelliği kullanırız. Karakterler yıldız şeklinde görünür.
- **reset:** Formdaki girilmiş tüm verileri temizler.
- **file:** Dosya yükleme butonudur. Bu butona tıklayınca bir pencere açılıp yüklenecek dosya buradan seçilecektir.
- **hidden:** Gizlemek istediğimiz kısımlar için bu özelliği kullanırız.
- **image:** Gönder butonu yerine resim koymak için bu özelliği kullanırız.
- **url:** Web adresi girişi için kullanılırız.
- **tel:** Telefon numarası girişi için kullanılırız.
- **number:** Sayı seçimleri için kullanırız. Aşağı yukarı oklar görünür.
- **range:** Sürgü ile minimum ve maximum değerler seçmek için kullanırız.
- **color:** Renk paletinden renk seçeriz.
- **search:** Arama motoru için bu metin kutusunu kullanırız.
- **email:** Mail adresi girişi için kullanılırız.
- **date:** Gün, ay ve yıl seçmemizi sağlar.
- **month:** Ay seçmemizi sağlar.
- **week:** Hafta seçmemizi sağlar.
- **time:** Saat ve dakika seçmemizi sağlar.
- **datetime-local:** Tek bir seferde gün, ay, yıl, saat ve dakika seçmemizi sağlar.
- **button:** Buton oluşturmak için bu özelliği kullanırız.

<input type="button">	Button
<input type="checkbox">	<input type="checkbox"/>
<input type="color">	<input type="color"/>
<input type="date">	dd-mm-yyyy <input type="date"/>
<input type="email">	farazc60@gmail.com
<input type="file">	Choose File No file chosen
<input type="hidden">	
<input type="image">	<input type="image"/>
<input type="number">	5 <input type="number"/>
<input type="password">	<input type="password"/>
<input type="radio">	<input type="radio"/>
<input type="range">	<input type="range"/>
<input type="reset">	Reset
<input type="submit">	Submit
<input type="text">	codewithfaraz
<input type="url">	https://www.codewithfaraz.com

# HTML Etiketleri

İsim:

John

Last Soyad:

"Güngör"

Gönder

```
<form action="/action_page.php">
```

```
  <label for="ad">İsim:</label><br>
```

```
  <input type="text" id="ad" name="ad" value="John"><br>
```

```
  <label for="soyad">Last Soyad:</label><br>
```

```
  <input type="text" id="soyad" name="soyad" value="Güngör"><br><br>
```

```
  <input type="submit" value="Gönder">
```

```
</form>
```

İsim:

John

Last Soyad:

"Güngör"

Gönder

**action:** Formun gönderileceği adresin yazıldığı parametredir. Eğer yukarıdaki kutuya bir şeyler yazıp gönder butonuna tıklarsanız, girdiğiniz bilgileri "kullanici.php" adlı veri dosyasına göndermiş olursunuz.

**method:** Formun gönderilme yöntemidir. "**GET**" metodu kullanıldığında istekler adres satırında görüntülenir. Bir de "**POST**" metodu vardır. Şimdilik bunlara değinmeyeceğiz.

**target="\_blank":** Form yeni bir pencerede açılır.

**value:** Butonun üstünde ne yazacağını belirler.

# HTML Etiketleri

- **VIDEO EKLEME:**

```
<video src="dosya.mp4" type="audio/mp4" width="320px"
height="240px" poster="foto.jpg" loop autoplay muted
controls> </video>
```

- **SES EKLEME:**

```
<audio src="bethoven.mp3" type="audio/mpeg" loop autoplay
muted controls> </audio>
```

# Link ve Kaynaklar

- <https://htmlcheatsheet.com/>
- Git eğitimi:  
<https://www.youtube.com/watch?v=rWG7oT7fePg>
- CSS
  - <https://css-tricks.com/>
  - <https://flexboxfroggy.com/#tr>
- <https://codepen.io/>

# PHP

## *~~Personal Home Page~~ Hypertext Preprocessor*

MBP 192 Internet Programlama

Bahar 25-26

iEU Bilgisayar Prog.





# PHP Nedir?

- HTML içine gömülebilen bir betik dilidir
- PHP Web Tabanlı ve Nesne yönelimli bir programlama dilidir.
- PHP büyük işletim sistemlerinde; **Linux**, birçok **Unix** türevi (HP-UX, Solaris, OpenBSD vb.), **Microsoft Windows**, **Mac OS**, çok çeşitli platformlarda çalışabilir.
- “Apache, IIS, nginx” gibi HTTP sunucularının büyük kısmını destekler.



# Neden saat değişti?

- index.html

```
<h1>Merhaba, bu HTML</h1>  
<p>Saat: ????? </p>
```

**Merhaba, bu HTML**

Saat: ?????

- index.php

```
<h1>Merhaba, bu PHP</h1>  
<p>Saat:  
<?php echo date('H:i:s'); ?>  
</p>
```

**Merhaba, bu PHP**

Saat: 21:24:30

**Kod:** php-ders-sunum-kodlari >> test.html, test.php

# Bilgisayarlarda PHP bulunmuyorsa – Online Compiler



- HTML ile düzgün çalışan:  
[https://www.w3schools.com/php/phptryit.asp?filename=tryphp\\_compiler](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_compiler)
- Alternatif:
  - <https://www.programiz.com/php/online-compiler/>
  - <https://onecompiler.com/php>



# Sistemde PHP kontrolü

- php -v : işletim sisteminde yüklü PHP versiyonu

```
C:\Users\RiverIsland>php -v
PHP 8.2.12 (cli) (built: Oct 24 2023 21:15:15) (ZTS Visual C++ 2019 x64)
Copyright (c) The PHP Group
Zend Engine v4.2.12, Copyright (c) Zend Technologies
```



# PHP Syntax ve Temel Kurallar

- **Kural 1:** PHP kodları **<?php .....**  
**?>** arasına yazılır

```
<?php
    // PHP kodu buraya
?>
```

- **Kural 2:** Her satır **noktalı virgül (;)** ile **biter**

```
<?php
    echo "Merhaba"; // ✓ Doğru
    echo "Dünya"    // ✗ Yanlış - noktalı virgül yok
?>
```

- **Kural 3:** PHP küçük/büyük harf duyarlıdır.

- **Kural 4:** PHP dosyaları **\*.php** uzantılı olmalı

- index.php ✓
- index.html ✗ (PHP çalışmaz)

- **Kural 5:** Kod yazımında yorum satırları kullanılır.  
Okunabilir kod!

```
<?php
    echo "Merhaba Dünya"; // Bu bir açıklama satırıdır.
    # Bu da başka bir tek satırlı yorum örneğidir.
    /* Bu bölümdeki kodlar,
       berber randevu sistemindeki
       veritabanı bağlantısını sağlar.
    */
    echo "Bağlantı başarılı.";
?>
```



# echo = Ekrana yazdır

## Ekrana

- "Merhab Dünya!"
- 5 + 3 toplamını
- H1 başlığı olarak  
"HTML de  
yazdırabilir"  
yazdıralım...

```
<?php
    echo "Merhaba Dünya";
    echo 5 + 3;
    echo "<h1>HTML de
yazdırabilir</h1>";
?>
```

**Kod:** php-ders-sunum-kodlari >> start.php



# HTML il PHP içi içe çalışması

## HTML İçine PHP Gömme

```
<!DOCTYPE html>
<html>
<head>
    <title>PHP</title>
</head>
<body>
    <h1>Merhaba MBP 192 SINIFI</h1>
    <p>
        <?php echo "Emrah Yüksel"; ?>
    </p>
    <p>
        Lorem ipsum...
        <?php echo " Bu kod HTML etiketi
            arasına eklendi "; ?>
        Excepteur sint...
    </p>
</body>
</html>
```

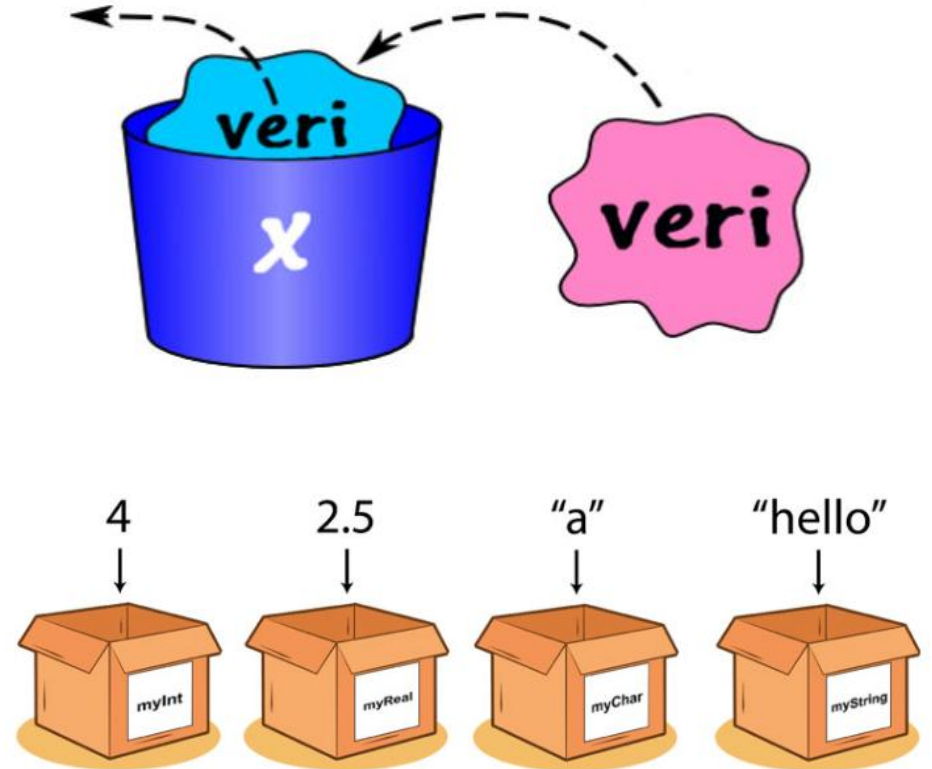
## PHP İçinden HTML Çıktısı Verme - " " dikkat

```
<!DOCTYPE html>
<html>
<head>
    <title>PHP</title>
</head>
<body>
    <?php
        echo "<h1>Merhaba MBP 192 </h1>";
        echo "<p> PHP Dersi </p>";
        echo "<p> Lorem ipsum...
            Bu kod HTML etiketi arasına
            eklendi
            Excepteur sint... </p>";
    ?>
</body>
</html>
```

**Kod:** php-ders-sunum-kodlari >> 2-html\_ic\_ice.php, 3-html\_ic\_ice.php

# Değişkenler

- Temek kullanım amacı, bir veriyi program boyunca tekrar tekrar yazmak yerine, ona bir isim atayıp o isimle çağırmaktır.
  - **Veri Yönetimi:** Değişkenin değerini bir kez değiştirdiğinizde, o değişkeni kullandığınız her yerde değer güncellenmiş olur
  - **Okunabilirlik:** Karmaşık veriler yerine anlamlı isimler (\$fiyat, \$musteri\_adi) kullanarak kodun ne yaptığı kolayca anlaşılır.



# Değişkenler ve Veri Tipleri

## Değişken Tanımlama Kuralları

- Tüm değişkenler **\$** işareti ile başlar sonra değişken ismi yazılır.
- Değişken adları bir harf veya alt çizgi ( \_ ) ile **başlamalıdır**.
- **Sayı ile başlayamazlar** ancak içinde sayı barındırabilirler.
- Büyük-küçük harfe duyarlıdır
  - ( \$ad ile \$AD farklı değişkenlerdir).

Değişken Adı	Durum	Neden?
\$randevuID	Uygun	Harf ile başlar, standart kurallara uyar.
\$_musteri	Uygun	Alt çizgi ile başlayabilir.
\$fiyat_2026	Uygun	İçinde sayı barındırabilir.
\$2randevu	Uygun Değil	Sayı ile başlayamaz.
\$toplam-tutar	Uygun Değil	Tire (-) karakteri içeremez (alt çizgi kullanılmalıdır).
\$ad soyad	Uygun Değil	Arada boşluk bırakılamaz.
\$değişken	Riskli	Türkçe karakter kullanımı teknik olarak çalışsa da önerilmez.

```

syntax error, unexpected integer "8", expecting variable or "{" or "$"
View Problem (Alt+F8) No quick fixes available
<?php $8 = 9; ?>
<?php echo $8; ?>
    
```

- **Kod:** php-ders-sunum-kodlari >> değişkenler.php



# Değişken Tanımlama Örnekleri

## Doğru

- **\$randevuID:** Harf ile başlar, standartlara uygundur.
  - `$randevuID = 101; echo "Randevu Numarası: " . $randevuID;`
- **\$\_musteri:** Alt çizgi ile başlama
  - `$_musteri = "Volkan"; echo "Müşteri Adı: " . $_musteri;`
- **\$fiyat\_2026:** İçinde sayı barındırabilir
  - `$fiyat_2026 = 250.00; echo "Gelecek Yıl Fiyatı: " . $fiyat_2026;`

## Hatalı

- **\$2randevu:** Sayı ile başlayamaz
  - `// HATALI KULLANIM $2randevu = "14:30"; // PHP bu satırda 'Parse error: syntax error, unexpected '2'' hatası verir.`
- **\$toplam-tutar:** PHP, tire işareti (-) çıkarma işlemi olarak algılar
  - `// HATALI KULLANIM $toplam-tutar = 500; // PHP burada '$toplam' değişkeninden 'tutar'ı çıkarmaya çalıştığını sanır ve hata verir.`
- **\$ad soyad:** Arada boşluk bırakılamaz.
  - `// HATALI KULLANIM $ad soyad = "Mustafa Volkan"; // PHP iki farklı kelime gördüğü için ne yapacağını bilemez.`
- **\$değişken:** Türkçe karakter kullanımı.(önerilmez)
  - `// RİSKLİ KULLANIM $ücret = 100; // Teknik olarak PHP 7+ sürümlerinde çalışır ancak dosya kodlaması (UTF-8) değişirse kodunuz bozulabilir. // Önerilen: $ucret = 100;`

**\$degisken\_adi = degisken\_degeri**



# Değişken Veri Tipleri ve Tanımlama

Değişken Yazım Standartları  
(CamelCase vs Snake\_Case)

Stil	Örnek
<b>CamelCase</b>	<code>\$randevuTarihi</code>
<b>Snake_Case</b>	<code>\$randevu_tarihi</code>

**var\_dump()**, bir değişkenin yapısı hakkında hem **değerini** hem de **türünü** (veri tipini) gösteren fonksiyondur

- **String (Metin):** Tırnak içerisinde yazılan her türlü metinsel veridir.
  - `$isim = "Volkan";`
- **Integer (Tam Sayı):** Ondalık kısmı olmayan pozitif veya negatif sayılardır.
  - `$yil = 2026;`
- **Float / Double (Ondalıklı Sayı):** Virgüllü (noktalı) sayılardır.
  - `$fiyat = 150.50;`
- **Boolean (Mantıksal):** Genellikle karar yapılarında kullanılır. Sadece iki değer alabilir:
  - `True` (doğru) veya `False` (yanlış).
- **Array (Dizi):** Tek bir değişken içinde birden fazla değer tutmamızı sağlar.
  - `$hizmetler = array("Saç Kesimi", "Sakal Tıraşı");`
- **Object (Nesne):** Dökümanlarda belirtilen "nesne yönelimli" yapının temelidir. Sınıflardan (class) türetilen verilerdir.
- **NULL:** Değişkenin hiçbir değer taşımadığını ifade eder.

- **Kod:** php-ders-sunum-kodlari >> veri\_tipleri.php

# Değişken Tanımlama ve Veri Tipleri



- `$hizmet_adi = "Saç ve Sakal Kesimi";`(String)
- `$sure_dakika = 45;` (Integer)
- `$ucret = 350.50;` (Float)
- `$kampanya_var_mi = true;` (Boolean)
- `$ekipmanlar = ["Makas", "Ustura", "Tarak"];`

```
$hizmet_adi = "Saç ve Sakal Kesimi"; //(String)
echo "<h3>$hizmet_adi = \"Saç ve Sakal Kesimi\";</h3>";
echo "Hizmet Adı: " . $hizmet_adi . "<br>";
var_dump($hizmet_adi);

$sure_dakika = 45; //(Integer)
echo "<h3>$sure_dakika = 45;</h3>";
echo "Hizmet Süresi: " . $sure_dakika . "<br>";
var_dump($sure_dakika);

$ucret = 350.50; //(Float)
echo "<h3>$ucret = 350.50;</h3>";
echo "Hizmet Ücreti: " . $ucret . "<br>";
var_dump($ucret);

$kampanya_var_mi = true; //(Boolean)
echo "<h3>$kampanya_var_mi = true;</h3>";
echo "Kampanya Var mı?: " . ($kampanya_var_mi ? "Evet" : "Hayır") . "<br>";
var_dump($kampanya_var_mi);

$ekipmanlar = ["Makas", "Ustura", "Tarak"]; //(Array)
echo "<h3>$ekipmanlar = [\"Makas\", \"Ustura\", \"Tarak\"];</h3>";
var_dump($ekipmanlar);
```

- **Kod:** php-ders-sunum-kodlari >> veri\_tipleri.php

- **Kod:** php-ders-sunum-kodlari >> veri\_tipleri.php



# Object (Nesne) veri tipi

- PHP'deki **Object (Nesne)** veri tipi, dökümanlarınızda vurgulanan "Nesne Yönelimli" (Object-Oriented) programlama yaklaşımının temelidir.
- Bir nesne, sadece bir değer değil; o değere ait **özellikleri (properties)** ve bu özellikler üzerinde işlem yapabilen **işlevleri (methods)** bir arada tutan kompleks bir yapıdır.
- Nesne veri tipini kullanabilmek için önce bir şablon olan **Sınıf (Class)** tanımlanmalı, ardından bu sınıftan bir **örnek (instance)** üretilmelidir.

```
<?php
// 1. Sınıf (Şablon) Tanımlama
class Berber {
    public $isim;
    public $uzmanlik;

    // Nesne oluşturulduğunda çalışan fonksiyon
    public function __construct($ad, $alan) {
        $this->isim = $ad;
        $this->uzmanlik = $alan;
    }

    public function selamVer() {
        return "Merhaba, ben " . $this->isim . ". Uzmanlık
alanım: " . $this->uzmanlik;
    }
}

// 2. Nesne (Object) Oluşturma
$berberNesnesi = new Berber("Volkan", "Saç Tasarımı");

// 3. Yapısını İnceleme
echo "<h2>Object Veri Tipi Yapısı:</h2>";
var_dump($berberNesnesi);

echo "<br><br>";
// Nesne içindeki metoda erişim
echo $berberNesnesi->selamVer();
?>
```

**Sınıf (Class):** Nesnenin nasıl olacağını belirleyen genel plandır

**Nesne (Object):** Bu plandan üretilmiş, hafızada yer kaplayan somut veridir.

**var\_dump Çıktısı:** Bu komutu kullandığınızda, PHP size değişkenin tipinin object olduğunu, hangi sınıfa ait olduğunu ve içindeki verileri detaylıca gösterir.

# Değişken Kapsamı (Scope)

- **Değişken kapsamı (variable scope)**, bir değişkenin kodun hangi bölümlerinden erişilebilir olduğunu belirleyen kurallar bütünüdür.
- Değişkenlerin nerede tanımlandığı, onların ömrünü ve görünürlüğünü doğrudan etkiler.
- Kapsam Tipleri:
  - Yerel Kapsam (Local Scope)
  - Global Kapsam (Global Scope)
  - Statik Kapsam (Static Scope)

- **Yerel Kapsam (Local Scope)**

- Bir fonksiyon içerisinde tanımlanan değişkenler "yerel" değişkendir. Bu değişkenlere sadece tanımlandıkları fonksiyonun içinden erişilebilir. Fonksiyon bittiğinde bu değişkenler bellekten silinir.

- **Global Kapsam (Global Scope)**

- Herhangi bir fonksiyonun dışında tanımlanan değişkenler "global" değişkendir. Bu değişkenlere fonksiyonların dışındaki her yerden erişilebilir. Ancak, bir fonksiyonun içinden global bir değişkene erişmek için global anahtar kelimesi kullanılmalıdır.
- **\$GLOBALS Superglobal** PHP, tüm global değişkenleri `$GLOBALS['degisken_adi']` isimli bir dizide saklar. Fonksiyon içinde global yazmak yerine bu diziye de kullanabilirsiniz.

- **Statik Kapsam (Static Scope)**

- Normalde bir fonksiyonun çalışması bittiğinde içindeki yerel değişkenler yok edilir. Ancak bir değişkenin değerini kaybetmemesini, fonksiyon her çağrıldığında kaldığı yerden devam etmesini istiyorsanız static anahtar kelimesini kullanırsınız.

- **Kod:** `php-ders-sunum-kodlari >> degisken_kapsam.php`

Kapsam Türü	Tanımlandığı Yer	Erişilebilirlik
<b>Yerel (Local)</b>	Fonksiyon içi	Sadece o fonksiyon içinden.
<b>Global</b>	Fonksiyon dışı	Fonksiyonlar hariç her yer (fonksiyon içinde <code>global</code> ile).
<b>Statik (Static)</b>	Fonksiyon içi	Sadece o fonksiyon içinden ama değeri sıfırlanmaz.

## Örnek

Adınızı, soyadınızı, yaşınızı, bölümünüzü farklı değişkenlere metin(string) olarak atayın.

Mezuniyet durumunuzu "mezun" isimli bir değişkene mantıksal(Boolean) olarak atayın.

Ekrana

Merhaba, ben ...ad soyad...

Yaşım: .....

Bölümüm: .....

Mezunsanız "Mezunoldum!"

Değilseniz "Hala öğrenciyim." olarak yazdırın

```
<?php
    $ad = "Ayşe";
    $soyad = "Yılmaz";
    $yas = 19;
    $universite = "İzmir Ekonomi
    Üniversitesi";
    $mezun = false;

    echo "Merhaba, ben $ad $soyad<br>";
    echo "Yaşım: $yas<br>";
    echo "Okulum: $universite<br>";
    if ($mezun) {
        echo "Mezun oldum!";
    }
    else {
        echo "Hala öğrenciyim.";
    }
?>
```

# OPERATÖRLER

- PHP'de operatörler, değişkenler ve değerler üzerinde işlemler gerçekleştirmek için kullanılan özel sembollerdir.

- Aritmetik Operatörler
- Atama Operatörleri
- Karşılaştırma Operatörleri
- Mantıksal Operatörler
- String (Metin) Operatörleri
- Artırma ve Azaltma Operatörleri
- Hata Operatörü

- **Kod:** php-ders-sunum-kodlari >> operatorler.php

# Aritmetik Operatörler

Sayısal değerlerle standart matematiksel işlemler yapmak için kullanılırlar.

Operatör	İsim	Örnek	Sonuç
+	Toplama	<code>\$a + \$b</code>	Toplam
-	Çıkarma	<code>\$a - \$b</code>	Fark
*	Çarpma	<code>\$a * \$b</code>	Çarpım
/	Bölme	<code>\$a / \$b</code>	Bölüm
%	Modül	<code>\$a % \$b</code>	Bölümden kalan
**	Üs Alma	<code>\$a ** \$b</code>	$a$ 'nın $b$ . kuvveti

```
<!DOCTYPE html>
<html>
<body>
  <h2>Matematiksel İşlemler</h2>
  <?php
    $a = 10;
    $b = 3;

    echo "A sayısı: $a, B sayısı: $b <br><br>";

    echo "Toplama ($a + $b): " . ($a + $b) . "<br>";
    echo "Çıkarma ($a - $b): " . ($a - $b) . "<br>";
    echo "Çarpma ($a * $b): " . ($a * $b) . "<br>";
    echo "Bölme ($a / $b): " . ($a / $b) . "<br>";
    echo "Modül ($a % $b): " . ($a % $b) . "<br>";
    echo "Üs Alma ($a ** $b): " . ($a ** $b) . "<br>";
  ?>
</body>
</html>
```

- Kod:**

- php-ders-sunum-kodlari >> operatorler.php
- php-ders-sunum-kodlari >> aritmetik\_operatorleri.php

# Atama Operatörleri

- Değişkenlere değer atamak için kullanılırlar.
- Bileşik atama operatörleri kodu kısaltmanıza yardımcı olur.

Operatör	Açıklama	Örnek İşlem
=	Atama İşlemi	<pre>&lt;?php \$adSoyad="Emrah Yüksel"; \$sayi=5; \$deger=TRUE; ?&gt;</pre>
+=	Toplayarak Atama	<pre>&lt;?php \$sayi=5; echo \$sayi+=10; // İşlem Sonucu 10 ?&gt;</pre>
-=	Çıkartarak Atama	<pre>&lt;?php \$sayi=5; echo \$sayi-=10; // İşlem Sonucu -5 ?&gt;</pre>
*=	Çarparak Atama	<pre>&lt;?php \$sayi=5; echo \$sayi*=10; // İşlem Sonucu 50 ?&gt;</pre>
/=	Bölerek Atama	<pre>&lt;?php \$sayi=5; echo \$sayi/=10; // İşlem Sonucu 0,5 ?&gt;</pre>
%	Mod Alarak Atama	<pre>&lt;?php \$sayi=5; echo \$sayi%=10; // İşlem Sonucu 5 ?&gt;</pre>
.=	Birleştirerek Atama	<pre>&lt;?php \$ad="Emrah"; echo \$ad.=" Yüksel"; // İşlem Sonucu Emrah Yüksel ?&gt;</pre>

- Kod:

- php-ders-sunum-kodlari >> operatorler.php
- php-ders-sunum-kodlari >> atama\_operatorleri.php

# Karşılaştırma Operatörleri

İki değeri karşılaştırmak için kullanılır ve sonuç olarak **true** veya **false** (boolean) döndürürler.

**==** : Eşittir (Sadece değer kontrolü).

**===** : Özdeşdir (Hem değer hem de **veri tipi** kontrolü).

**!=** veya **<>** : Eşit değildir.

**!==** : Özdeş değildir.

**>**, **<**, **>=**, **<=** : Büyüklük/Küçüklük kontrolleri.

**<=>** (**Spaceship**): Sol taraf küçükse **-1**, eşitse **0**, büyükse **1** döndürür.

Operatör	Açıklama	Örnek İşlem
<b>==</b>	Eşittir	<code>\$a == \$b</code>
<b>===</b>	Denktir	<code>\$a === \$b</code>
<b>!=</b>	Eşit değildir	<code>\$a != \$b</code>
<b>!==</b>	Denk değildir	<code>\$a !== \$b</code>
<b>&lt;&gt;</b>	Eşit değildir	<code>\$a &lt;&gt; \$b</code>
<b>&lt;</b>	Küçüktür	<code>\$a &lt; \$b</code>
<b>&gt;</b>	Büyüktür	<code>\$a &gt; \$b</code>
<b>&lt;=</b>	Küçük ve eşittir	<code>\$a &lt;= \$b</code>
<b>=&gt;</b>	Büyük ve eşittir	<code>\$a =&gt; \$b</code>

- Kod:

- `php-ders-sunum-kodlari >> operatorler.php`
- `php-ders-sunum-kodlari >> karsilastirma_operatorleri.php`

# Mantıksal Operatörler

- Birden fazla koşulu birleştirmek için kullanılır.
- ÖRNEK:** Berber randevu sisteminizde
  - «Eğer kullanıcı giriş yapmışsa **VE** admin ise»

**&& (and):** Ve (Her iki koşul da doğru olmalı).

**|| (or):** Veya (En az bir koşul doğru olmalı).

**! (not):** Değil (Koşulun tersini alır).

Operatör	Açıklama	Örnek İşlem
<code>\$a and \$b</code>	Ve	<code>\$a</code> ve <code>\$b</code> her ikisi de doğruysa sonuç doğrudur (TRUE).
<code>\$a or \$b</code>	Veya	<code>\$a</code> veya <code>\$b</code> doğruysa sonuç doğrudur.
<code>\$a xor \$b</code>	Ayrıcalıklı Veya	<code>\$a</code> veya <code>\$b</code> doğruysa sonuç doğru, her ikiside doğruysa sonuç yanlıştır (FALSE).
<code>! \$a</code>	Değil	<code>\$a</code> doğru değilse sonuç doğrudur.
<code>\$a &amp;&amp; \$b</code>	Ve	<code>\$a</code> ve <code>\$b</code> her ikisi de doğruysa sonuç doğrudur.
<code>\$a    \$b</code>	Veya	<code>\$a</code> veya <code>\$b</code> doğruysa sonuç doğrudur.

- Kod:
  - php-ders-sunum-kodlari >> operatorler.php
  - php-ders-sunum-kodlari >> mantiksal\_operatorler.php

# Artırma ve Azaltma Operatörleri

Değişken değerini bir artırmak veya azaltmak için kullanılır.

`++$x` : Önce artır, sonra işlem yap.

`$x++` : Önce işlem yap, sonra artır.

`--$x` / `$x--` : Azaltma işlemleri.

```
$a = 5;
echo ++$a; // Önce 6 yapar, sonra ekrana 6 yazar.
```

```
$a = 5;
echo $a++; // Önce ekrana 5 yazar, işlem bitince arka planda $a'yı 6 yapar.
echo $a;    // Şimdi yazdırırsak sonuç 6 olur.
```

- Kod:
  - php-ders-sunum-kodlari >> operatorler.php
  - php-ders-sunum-kodlari >> arttirma-azaltma-operatoru.php
  - php-ders-sunum-kodlari >> arttirma-azaltma-operatorleri.php

# Artırma ve Azaltma Operatörleri

## KOD:

php-ders-sunum-kodlari >>  
arttirma\_azaltma\_operatorleri.php

```
<?php
$sayi = 10;
echo "Başlangıç: " . $sayi . "<br>"; // 10
// Önce artırma akışı
echo "Önce artır (++$sayi): " . ++$sayi . "<br>"; //
Ekranı 11 görürüz.
// Sonradan artırma akışı
echo "Sonra artır ($sayi++): " . $sayi++ . "<br>"; //
Ekranı hala 11 görürüz (çünkü işlemden sonra artacak).
echo "Artırma sonrası son durum: " . $sayi . "<br>"; //
Şimdi 12 olduğunu görürüz.
// Azaltma akışı
echo "Önce azalt (--$sayi): " . --$sayi; // Ekranı 11
görürüz.
?>
```

Örnek Kod	İşlem Akışı	Sonuçlar
<code>\$y = ++\$x;</code>	<ol style="list-style-type: none"> <li>1. <code>\$x</code> artar</li> <li>2. Yeni <code>\$x</code>, <code>\$y</code> 'ye atanır.</li> </ol>	<code>\$x</code> ve <code>\$y</code> aynı olur.
<code>\$y = \$x++;</code>	<ol style="list-style-type: none"> <li>1. Mevcut <code>\$x</code>, <code>\$y</code> 'ye atanır</li> <li>2. <code>\$x</code> artar.</li> </ol>	<code>\$y</code> eski değerinde kalır, <code>\$x</code> artar.

# String (Metin) Operatörleri

Metinleri birleştirmek için kullanılır. PHP'nin **"HTML içine gömülebilen"** yapısında çıktı üretirken sıkça kullanılır.

`.` : Birleştirme operatörü.

PHP

```
$ad = "Emrah";
$soyad = "Yüksel";
echo $ad . " " . $soyad; // Çıktı: Emrah Yüksel
```

```
<?php
$mesaj = "Merhaba";
$mesaj .= " Dünya!"; // $mesaj = $mesaj . " Dünya!" işleminin kısa halidir.
echo $mesaj; // Çıktı: Merhaba Dünya!
?>

<?php
$ders_kodu = "MBP 192";
echo "<h1>" . $ders_kodu . " İnternet Programlama</h1>";
?>
```

- **Kod:**
  - php-ders-sunum-kodlari >> operatorler.php
  - php-ders-sunum-kodlari >> metin\_operatorleri.php

# SORU

- Aşağıdaki PHP kod bloğu çalıştırıldığında ekrana ne yazdırılır?

```
<?php
$sayi = 10;
$sayi += 5;
$sonuc = $sayi % 4;
echo "İşlem Sonucu: " . $sonuc;
?>
```

# SORU

- Aşağıdaki PHP kod bloğu çalıştırıldığında ekrana ne yazdırılır? Beklenen çıktı nedir?

```
<?php
$puan = 50;
$ad = "Öğrenci";
echo $ad . " puanı: " . $puan++;
echo " | Yeni durum: " . $puan;
?>
```

# SORU

- Bir randevu sisteminde, aşağıdaki koşulun true (doğru) dönmesi için hangi durumun gerçekleşmesi gerekir?

```
<?php
if ($giris_yapildi == true && ($yetki == "admin" || $ucret <= 0)) {
    echo "Erişim başarılı.";
}
?>
```

- **A)** Kullanıcı giriş yapmamışsa ama admin ise.
- **B)** Kullanıcı giriş yapmışsa VE (ya admin ise YA DA ücret 0 ve altındaysa).
- **C)** Ücreti 0 olan herkes giriş yapabilir.
- **D)** Sadece adminler giriş yapabilir.

# Tür dönüşümleri (Type casting)

## Tür Dönüşümü Akışı ve Dikkat Edilmesi Gerekenler

- Dönüşüm işlemleri yapılırken veri kaybı yaşanabileceği unutulmamalıdır
  - Veritabanından gelen tüm veriler genellikle "string" olarak döner.
    - Bu verileri matematiksel işlemlerde kullanmadan önce **(int)** veya **(float)** ile dönüştürmek, uygulamanızın daha kararlı çalışmasını sağlar.
- **Otomatik Tür Dönüşümü (Type Juggling)**
    - işlem sırasında bir değişkenin türünü o anki bağlama göre kendisi belirler.
  - **Manuel Tür Dönüşümü (Type Casting):**
    - Geliştiricinin, bir değişkenin türünü açıkça belirtmesi işlemidir.
    - Değişkenin önüne parantez içinde hedef tür yazılarak gerçekleştirilir.
  - **Özel Dönüşüm Fonksiyonları:**
    - Daha kontrollü dönüşümler yapmak için PHP'nin sunduğu hazır fonksiyonlardır

# Otomatik Tür Dönüşümü (Type Juggling)

İşlem sırasında bir değişkenin türünü o anki bağlama göre kendisi belirler.

```
$sayi = 5;           // Integer
$metin = "10";       // String
$sonuc = $sayi + $metin; // Sonuç: 15 (Integer)
```

```
echo "<h3>Tam Sayıya Dönüştürme (int) veya (integer)</h3><br>";
$fiyat = 150.99;
$tamFiyat = (int)$fiyat;
// Sonuç: 150 (Veri kaybı yaşanır, .99 kısmı silinir)

$metinSayi = "42isik";
$sayi = (int)$metinSayi;
// Sonuç: 42 (Metnin başındaki sayıyı alır)
$metinSayi2 = "isik42";
$sayi2 = (int)$metinSayi2;
// Sonuç: 0 (Metnin başında sayı olmadığı için sıfır olur)
echo "Fiyat: " . $fiyat . " → Tam Fiyat: " . $tamFiyat; // Fiyat: 150.99 → Tam Fiyat: 150
echo "<br>";
echo "Metin Sayı: " . $metinSayi . " → Sayı: " . $sayi; // Metin Sayı: 42isik → Sayı: 42
echo "<br>";
echo "Metin Sayı 2: " . $metinSayi2 . " → Sayı 2: " . $sayi2; // Metin Sayı 2: isik42 → Sayı 2: 0
echo "<br>";
```



# Manuel Tür Dönüşümü (Type Casting)

Dönüşüm Türü	Kullanım	Açıklama
<b>(int), (integer)</b>	<code>(int)\$degisken</code>	Tam sayıya dönüştürür.
<b>(bool), (boolean)</b>	<code>(bool)\$degisken</code>	Mantıksal değere (true/false) dönüştürür.
<b>(float), (double)</b>	<code>(float)\$degisken</code>	Ondalıklı sayıya dönüştürür.
<b>(string)</b>	<code>(string)\$degisken</code>	Metne (karakter dizisine) dönüştürür.
<b>(array)</b>	<code>(array)\$degisken</code>	Diziye dönüştürür.
<b>(object)</b>	<code>(object)\$degisken</code>	Nesneye dönüştürür.

- Kod: php-ders-sunum-kodlari >> tur\_donusumu.php

# Özel Dönüşüm Fonksiyonları



Verileri daha kontrollü ve fonksiyonel bir şekilde dönüştürmek için özel fonksiyonlar kullanılır.

- **settype():**
  - değişkenin türünü kalıcı olarak değiştirir.

Description
<code>settype(mixed \$var, string \$type): bool</code>
Set the type of variable <b>var</b> to <b>type</b> .
Parameters
<b>var</b> The variable being converted.
<b>type</b> Possibles values of <b>type</b> are: <ul style="list-style-type: none"><li>◦ "boolean" or "bool"</li><li>◦ "integer" or "int"</li><li>◦ "float" or "double"</li><li>◦ "string"</li><li>◦ "array"</li><li>◦ "object"</li><li>◦ "null"</li></ul>
Return Values
Returns <b>true</b> on success or <b>false</b> on failure.

<https://www.php.net/manual/en/function.settype.php>

## Değer Döndüren Fonksiyonlar (intval, floatval, strval)

- Bu fonksiyonlar değişkenin değerini alır, istenen tipe çevirir ve sonucu döndürür; ancak orijinal değişkene dokunmazlar
  - **intval(\$degisken):**  
Değişkenin tam sayı değerini döndürür.
  - **floatval(\$degisken):**  
Değişkenin ondalıklı sayı değerini döndürür.
  - **strval(\$degisken):**  
Değişkenin metin (string) değerini döndürür.



# Özel Dönüşüm Fonksiyonları

```
intval(mixed $value, int $base = 10): int
```

Returns the [int](#) value of **value**, using the specified **base** for the conversion (the default is base 10). `intval()` should not be used on objects, as doing so will emit an [E\\_WARNING](#) level error and return 1.

## Parameters

### value

The scalar value being converted to an integer

### base

The base for the conversion

#### Note:

If **base** is 0, the base used is determined by the format of **value**:

- if string includes a "0x" (or "0X") prefix, the base is taken as 16 (hex); otherwise,
- if string starts with a "0b" (or "0B"), the base is taken as 2 (binary); otherwise,
- if string starts with "0", the base is taken as 8 (octal); otherwise,
- the base is taken as 10 (decimal).

## Return Values

The integer value of **value** on success, or 0 on failure. Empty arrays return 0, non-empty arrays return 1.

The maximum value depends on the system. 32 bit systems have a maximum signed integer range of -2147483648 to 2147483647. So for example on such a system, `intval('1000000000000')` will return 2147483647. The maximum signed integer value for 64 bit systems is 9223372036854775807.

Strings will most likely return 0 although this depends on the leftmost characters of the string. The common rules of [integer casting](#) apply.

## Değer Döndüren Fonksiyonlar (intval, floatval, strval)

- Bu fonksiyonlar değişkenin değerini alır, istenen tipe çevirir ve sonucu döndürür; ancak orijinal değişkene dokunmazlar
  - **intval(\$degisken):**  
Değişkenin tam sayı değerini döndürür.
  - **floatval(\$degisken):**  
Değişkenin ondalıklı sayı değerini döndürür.
  - **strval(\$degisken):**  
Değişkenin metin (string) değerini döndürür.

# SORU??

Aşağıdaki PHP kod bloğu çalıştırıldığında tarayıcı ekranında (çıktıda) ne görünür? Lütfen nedenlerini açıklayarak belirtiniz.

```
<?php
$gelen_ucret = "350"; // Formdan gelen veri (String)
$hizmet_bedeli = 350; // Sistemdeki sabit değer (Integer)

echo "1. Sonuç: ";
var_dump($gelen_ucret == $hizmet_bedeli);

echo "<br>2. Sonuç: ";
var_dump($gelen_ucret === $hizmet_bedeli);
?>
```

# Ödev 1

- PHP fonksiyonlarında echo ve print nasıl çalışır, farkı nedir?
- Metin/Text veri tipinin "" (çift tırnak) veya ' (tek tırnak) içinde yazılmasını farkları nelerdir?
- Metin yazarken "" (çift tırnak) içinde çift tırnak nasıl kullanılır? Yok sayma (escape) kaçış işareti.

# Sabitler

Programın çalışması boyunca değeri değiştirilemeyen isimlerdir.

**Sabit Tanımlama:**

**define()** Fonksiyonu ile veya **const** ile tanımlanır.

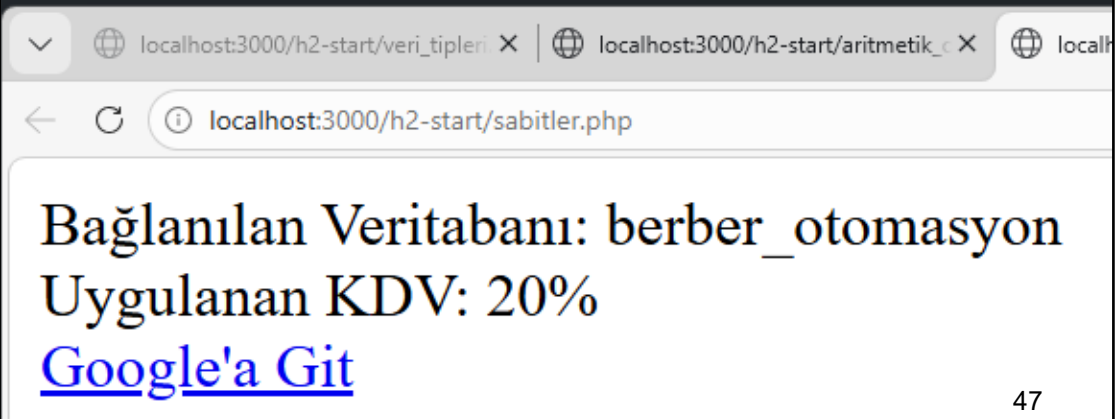
- **Dolar İşareti Yoktur:** Değişkenlerin aksine sabitler \$ ile başlamaz.
- **Global Kapsam:** Sabitler tanımlandıkları andan itibaren tüm sayfa ve fonksiyonlar içerisinde otomatik olarak "global" kapsamdadır
- **Değiştirilemezlik:** Program akışında bir sabite yeni bir değer atamaya çalışmak hata almanıza neden olur.
- **Yazım Kuralı:** genellikle tamamı **BÜYÜK HARF** ile yazılır.

```
<!DOCTYPE html>
<html>
<body>
    <?php
        // define("SABİT_ADI", "değer");
        define("VERITABANI", "berber_otomasyon");
        define("KDV_ORANI", 0.20);

        const SITE_URL = "https://www.google.com";

        echo "Bağlanılan Veritabanı: " . VERITABANI . "<br>";
        echo "Uygulanan KDV: " . (KDV_ORANI * 100) . "% <br>";

        echo "<a href='" . SITE_URL . "'>Google'a Git</a>";
    ?>
</body>
</html>
```



**KOD:** php-ders-sunum-kodlari >> sabitler.php

## SORU

Aşağıdaki  
kod  
bloğunda  
yapılan  
hata nedir?

```
<?php  
define("PI", 3.14);  
$yari_cap = 5;  
PI = 3; // Hatalı satır  
echo PI * ($yari_cap ** 2);  
?>
```

PHP'de sabitler bir kez tanımlandıktan sonra değerleri değiştirilemez. `PI = 3;` satırı bir sabit üzerinde yeniden atama yapmaya çalıştığı için **Parse Error** (Sözdizimi Hatası) verecektir.



# Sihirli Sabitler (Magic Constants)

Yazıldıkları yere göre değerleri değişen özel sabitlerdir.

Normal sabitlerin aksine, PHP yorumlayıcısı tarafından çalışma anında otomatik olarak atanırlar.

```
<!DOCTYPE html>
<html>
<body>
  <h2>PHP Sihirli Sabitler Testi</h2>
  <?php
    echo "Bu kodun bulunduğu satır: " . __LINE__ . "<br>"; //
    echo "Dosya konumu: " . __FILE__ . "<br>"; //

    function testFonksiyonu() {
        echo "Şu an çalışan fonksiyon: " . __FUNCTION__; //
    }

    testFonksiyonu();
  ?>
</body>
</html>
```

Bu sabitler her zaman  
**iki alt çizgi (\_\_) ile**  
**başlar ve biter**

**KOD:** php-ders-sunum-kodlari >> sihirli\_sabitler.php

# Değişken, Sabit ve Sihirli Sabit Farkı



- **Değişken (\$puan):** Çalışma anında değeri değişebilir, dolar işaretiyle tanımlanır.
- **Sabit (define):** Değeri asla değişmez, uygulama genelinde (global) erişilebilirdir.
- **Sihirli Sabit (\_\_LINE\_\_):** Değeri, kodun hangi dosyada veya hangi satırda olduğuna göre PHP tarafından otomatik değiştirilir.

## SORU

Aşağıdaki  
kodun çıktısı  
nedir,  
açıklayınız?

```
1  <!DOCTYPE html>
2  <html>
3  <body>
4      <h2>PHP Sihirli Sabitler</h2>
5      <?php
6          echo __LINE__;
7          echo "<br>";
8          echo __LINE__;
9      ?>
10 </body>
11 </html>
```



# FORMLAR

Bir web sayfasında kullanıcıdan bilgi alabilmek için <form> etiketine ihtiyaç duyarız.

**method / action / name**

```
<form method="POST" action="formlar_baslangic_POST.php">
  <label>1. Sayı:</label><br>
  <input type="number" name="sayi1" required><br>
  <label>2. Sayı:</label><br>
  <input type="number" name="sayi2" required><br><br>
  <input type="submit" value="Topla">
</form>
```

# FORM: method/action/name

- **method** = "POST" / "GET"
  - Verinin sunucuya nasıl gönderileceğini belirler.
  - "**POST**" yöntemi, verileri URL'nin içinde değil, HTTP isteğinin gövdesinde (arka planda) gizli bir şekilde gönderir
  - "**GET**" yönetimi, verileri URL'nin içinde iletir ve değerler görünür.
- **action**="\*\*\*\*.php";
  - formun içindeki veriler "Gönder" (Submit) butonuna basıldığında **bu verilerin hangi adrese (URL) gönderileceğini** belirler.
  - Eğer bu niteliği boş bırakırsanız veya hiç yazmazsanız, form verileri **bulunduğunuz sayfanın kendisine** tekrar gönderilir.



# FORM: method/action/name

- **name** <input name="sayi1">
  - **name değeri**, PHP tarafında veriyi yakalamak için kullandığımız "anahtar" ismidir.
  - Kullanıcı formu gönderdiğinde, tarayıcı bir veri paketi oluşturur. Bu paket şu mantıkla çalışır:  
***name\_değeri = kullanıcının\_yazdığı\_değer***
  - Eğer bir input elemanına name vermezseniz, o kutuya ne yazılırsa yazılsın PHP tarafına **hiç** iletmez.

# Method / Action / Name ilişkisi



- **action** verinin **nereye** gideceğini belirlerken,
- **method** (GET veya POST) verinin **nasıl** (açıktan mı yoksa gizli mi) gideceğini belirler.
- **name** gönderilecek verinin form yapısında nereden alınacağını belirtir.

Özellik	HTML ( <code>name</code> )	PHP Karşılığı
Görevi	Veriye kimlik/isim verir.	Kimliği kullanarak veriyi okur.
Yazım Şekli	<code>&lt;input name="eposta"&gt;</code>	<code>\$_POST['eposta']</code>
Eksik Olursa	Veri sunucuya gönderilmez.	<code>Undefined index</code> hatası alırsınız.



# Alınan değerin Tür dönüşümü

- Formlardan gelen veriler varsayılan olarak **"string" (metin)** gelir.
- Tür dönüşümü ile kontrol edilmesi veya tür dönüşümü yapılması, çalışma açısından önemlidir.
- PHP, aritmetik işlem sırasında **Otomatik Tür Dönüşümü** yaparak bunları sayıya çevirir.

**KOD:** php-ders-sunum-kodlari >>formlar\_baslangic\_POST.php  
php-ders-sunum-kodlari >> formlar\_baslangic\_GET.php

# FORM süreç



```
<form method="GET" action="formlar_baslangic_GET.php">
  <label>1. Sayı1:</label><br>
  <input type="number" name="sayi1" required><br>
  <label>2. Sayı1:</label><br>
  <input type="number" name="sayi2" required><br><br>

  <input type="submit" value="Topla">
</form>
<?php
// Formun gönderilip gönderilmediğini kontrol edelim
if (isset($_GET['sayi1']) && isset($_GET['sayi2'])) {

    // Formdan gelen verileri değişkenlere atayalım
    $s1 = $_GET['sayi1'];
    $s2 = $_GET['sayi2'];

    // Aritmetik toplama işlemi
    $toplam = $s1 + $s2;

    // Sonucu HTML içine birleştirerek yazdıralım
    echo "<h3>İşlem Sonucu</h3>";
    echo "<p>$s1 + $s2 = <b>$toplam</b></p>";
}
?>
```

**KOD:** php-ders-sunum-kodlari >>formlar\_baslangic\_POST.php  
php-ders-sunum-kodlari >> formlar\_baslangic\_GET.php



# FORM ÖRNEK

## POST

```
<!DOCTYPE html>
<html>
  <head><title>Formlar ve POST Metodu</title></head>
  <body>
    <form method="" action="formlar_baslangic_POST.php">
      <label>1. Sayı:</label><br>
      <input type="number" name="sayi1" required><br>
      <label>2. Sayı:</label><br>
      <input type="number" name="sayi2" required><br><br>

      <input type="submit" value="Topla">
    </form>
    <?php
      // Formun gönderilip gönderilmediğini kontrol edelim
      if (isset($_POST['sayi1']) && isset($_POST['sayi2'])) {

        // Formdan gelen verileri değişkenlere atayalım
        $s1 = $_POST['sayi1'];
        $s2 = $_POST['sayi2'];

        // Aritmetik toplama işlemi
        $toplam = $s1 + $s2;

        // Sonucu HTML içine birleştirerek yazdıralım
        echo "<h3>İşlem Sonucu</h3>";
        echo "<p>$s1 + $s2 = <b>$toplam</b></p>";
      }
    <?>

  </body>
</html>
```

## GET

```
<!DOCTYPE html>
<html>
  <head><title>Formlar ve GET Metodu</title></head>
  <body>
    <form method="GET" action="formlar_baslangic_GET.php">
      <label>1. Sayı:</label><br>
      <input type="number" name="sayi1" required><br>
      <label>2. Sayı:</label><br>
      <input type="number" name="sayi2" required><br><br>

      <input type="submit" value="Topla">
    </form>
    <?php
      // Formun gönderilip gönderilmediğini kontrol edelim
      if (isset($_GET['sayi1']) && isset($_GET['sayi2'])) {

        // Formdan gelen verileri değişkenlere atayalım
        $s1 = $_GET['sayi1'];
        $s2 = $_GET['sayi2'];

        // Aritmetik toplama işlemi
        $toplam = $s1 + $s2;

        // Sonucu HTML içine birleştirerek yazdıralım
        echo "<h3>İşlem Sonucu</h3>";
        echo "<p>$s1 + $s2 = <b>$toplam</b></p>";
      }
    <?>

  </body>
</html>
```