Burdur Mehmet Akif Ersoy Üniversitesi

MAKÜ | GMYO
GÖLHİSAR MESLEK YÜKSEKOKULU

İnternet Programcılığı I

Öğr. Gör. Hüseyin Şengün
hsengun@mehmetakif.edu.tr

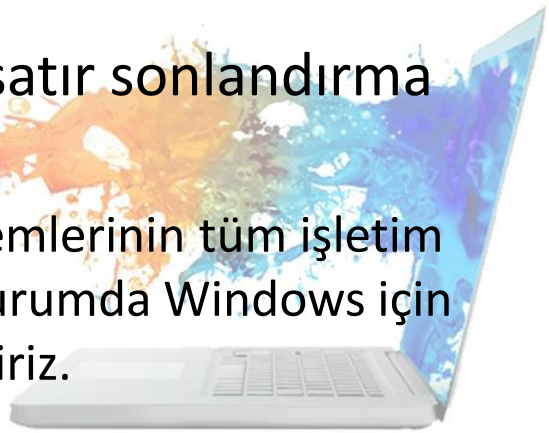
7. Hafta
Dosya İşlemleri

Dosya İşlemleri

- PHP ile geliştirilen uygulamalarda genellikle veri tabanlarında bilgi saklanmasına karşın bazı durumlarda fiziksel dosyalar üzerine de bilgi yazmak gerekebilir.
- Bu durumların en yaygın olanı işlem kayıtlarının tutulduğu “log” dosyalarıdır. “Log” dosyaları, genellikle normal metin editörleri tarafından okunabilen satırlardan oluşmaktadır.
- Kayıtların bir fiziksel dosya üzerinde saklanması veya fiziksel bir dosyadaki bilginin okuması PHP ile mümkündür.

Dosya İşlemleri

- Dosya işlemlerinde en çok dikkat edilmesi gereken önemli unsurlardan birisi, satır sonlarıdır. Windows, Linux ve MacOS işletim sistemi aileleri satır sonlandırmak için farklı ifadeler kullanmaktadır.
- Linux işletim sistemlerinde satır sonlandırmak için kullanılan karakter **“\n”**’dir.
- MacOS işletim sistemleri için satır sonlandırma karakteri ise **“\r”**’dir.
- Aynı şekilde Windows işletim sistemleri içinse satır sonlandırma karakteri **“\r\n”**’dir.
- Eğer yazdığımız metin dosyalarının satır sonlandırma işlemlerinin tüm işletim sistemlerinde aynı görüntüyü vermesini istiyorsak; bu durumda Windows için geçerli olan satır sonlandırma ifadesi “\r\n”’yi kullanabiliriz.



Dosya İşlemleri

- Satır sonlandırmak için ayrıca **PHP_EOL** değişkenini de kullanabiliriz. Bu değişken sunucunun işletim sisteminin satır sonlandırma ifadesini içermektedir.
- PHP dilinde dosya içeriğinin okunması, tekrar yazılması ve dosya oluşturulması birden fazla yöntem ile gerçekleştirilebilir.
- Bu yöntemler aşağıdaki gibi listelenebilir.
- **file_get_contents / file_put_contents**
- **fopen / fwrite**

Dosya İşlemleri

- **Dosya İçeriğinin Yazılması**
- Sunucu sabit diski üzerinde bulunan bir dizinde dosya oluşturmak için öncelikle dizininin yazılabilir olması gerekmektedir. Varsayılan olarak dizinlerde sadece okuma yetkisi bulunmaktadır.
- Bu durumun nedeni ise istenmeyen dosya oluşturmalara karşı işletim sistemlerinin aldığı güvenlik önlemidir.
- ***Yazılabilirlik kontrolü***
- PHP dilinde bir dizininin yazılabilir olduğunu kontrol etmek için kullanılabilecek fonksiyon mevcuttur. Kontrol fonksiyonu olan "**is_writable**"'ın kullanımı için aşağıdaki örneği inceleyebilirsiniz.

Dosya İşlemleri

- Örnek olarak sunucunun kök dizininde (*C:\xampp\htdocs\phpdersleri\arsiv*), “arsiv” adında bir klasör içinde “**aof.txt**” dosyası oluşturalım.
- Bu durumda sunucu kök dizininin Windows Gezginindeki görüntüsü aşağıdaki gibi olacaktır.

- Resimde görünen “**arsiv**" klasörünün içi boş olabilir.
- Klasörün yazılabilir olduğunun kontrolünü ise “**ata-aof.php**” betiği ile denetleyeceğiz. Bu durumda “**ata-aof.php**” dosyasının içeriği ve tarayıcıda oluşacak ekran görüntüsü aşağıdaki gibi olabilir.

Dosya İşlemleri

- "**arsiv**" klasörünün yazılabilir olduğunun kontrolü

```php
<?php
$hedefDizin = "arsiv";
$yetkiDizin = is_writable( $hedefDizin );

var_dump( $yetkiDizin );
?>
```

- Tarayıcıda oluşacak ekran görüntüsü

- fonksiyon cevap olarak bize "**true**" değerini göndermektedir. Bu durumda "arsiv" klasörünün yazılabilir olduğunu anlayabiliriz.

Dosya İşlemleri

- Eğer yazılabilir olmasaydı aşağıdaki PHP kodunu kullanarak klasörü yazılabilir hale getirebilirdik.

```php
<?php
$hedefDizin = "arsiv";
$yetkiDizin = is_writable( $hedefDizin );

if( $yetkiDizin == false )
{
    chmod( $yetkiDizin, 777 );
}
?>
```

Klasörü yazılabilir olarak değiştirme

Dosya İşlemleri

- **Metot: file_put_contents**
- Sabit disk üzerinde bir dosya oluşturmak veya varolan bir dosyaya eklemeler yapmak için kullanılabilecek en basit PHP fonksiyonu **file_put_contents**' dir.
- Bu fonksiyon, 2. metotta görülecek olan işlemlerin birleştirilmiş şeklidir.
- PHP dilinde **file_put_contents** fonksiyonu yazılacak dosyanın varlığını (mevcut olup olmadığını) kendisi araştırır. Eğer dosya mevcut ise var olan dosyaya yeni veriyi ekler; dosya mevcut değilse, oluşturarak veriyi içine yazar.
- Fonksiyonun parametre olarak aldığı değerler aşağıdaki gibi verilebilir.

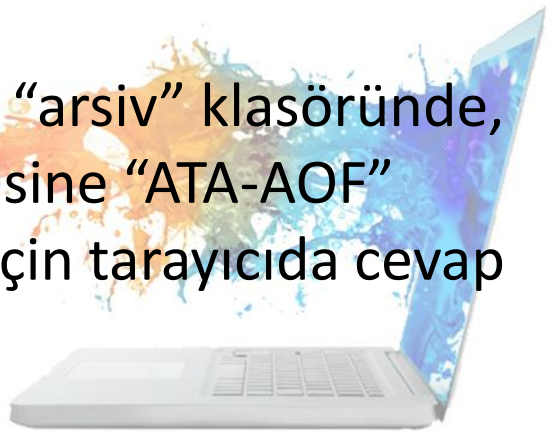
1.Dosya ismi *2.Veri (yazılacak metin)* *3.Seçenekler*

Dosya İşlemleri

- **file_put_contents** fonksiyonu kullanarak bir dosya içerisine veri yazmak için aşağıdaki örnek kullanılabilir.

```php
<?php
$bayt = file_put_contents("arsiv/aof.txt", "ATA-AOF");

var_dump( $bayt );
?>
```

- Fonksiyon cevap olarak, dosya yazma işlemi başarılı olursa, yazılan verinin bayt değerini; başarısız olursa “false” değerini geri gönderecektir.
- Örneğimizde; yazma işlemi başarılı olduğu için; “arsiv” klasöründe, “aof.txt” adında bir dosya oluşturulmuş ve içerisine “ATA-AOF” metni yazılmıştır. Her karakter bir bayt olduğu için tarayıcıda cevap olarak “7” değeri görüntülenebilir.



Dosya İşlemleri

- **Metot: fwrite**
- PHP dilinde bir dosya içerisine veri yazmak için kullanılabilecek en kapsamlı fonksiyon "**fwrite**"'dir. Bu fonksiyon, bir dosya üzerinde işlem yapmak için doğrudan kullanılamaz.
- Dosya içeriğine veri yazabilmek için aşağıdaki sıralama ile fonksiyonların çalıştırılması gerekmektedir.

  1. fopen
  2. fwrite
  3. fclose

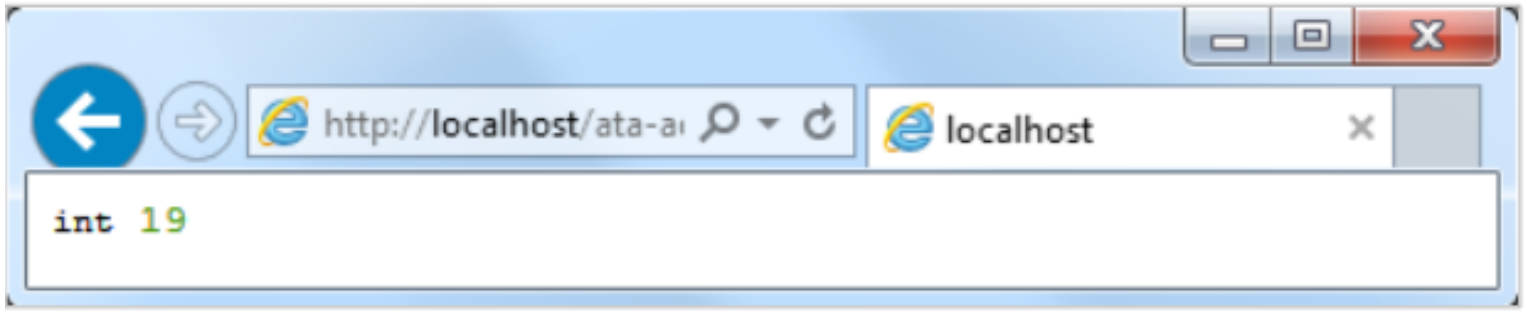
Dosya İşlemleri

- Yukarıda sıralanan fonksiyonlar ile öncelikle yazılacak dosya açılmalı, daha sonra içeriği yazılmalı ve dosya tekrar kapatılmalıdır.
- Bir önceki konuda yer alan örneğin aynısını 2. metot ile uygulayacak olursak; PHP kodları, tarayıcı ekran görüntüsü ve oluşturulan dosyanın içeriği aşağıdaki gibi olacaktır.

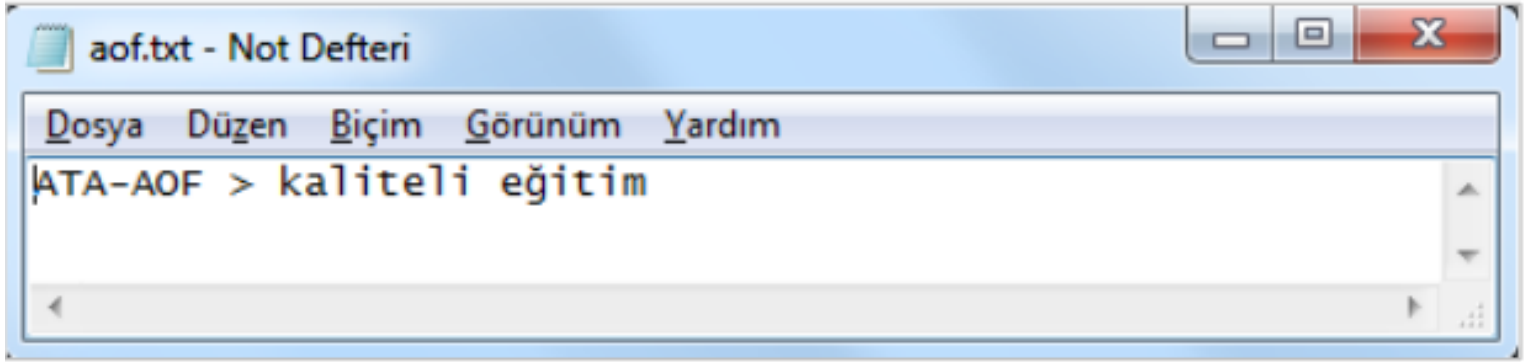
```php
<?php
$bayt = file_put_contents("arsiv/aof.txt", "ATA-AOF");

$dosya = fopen( "arsiv/aof.txt", "a+" );
$yazim = fwrite( $dosya, " > kaliteli eğitim" );
$kapat = fclose( $dosya );

var_dump( $yazim );
?>
```

- 2. Metot ile dosyaya yazdırma

Dosya İşlemleri



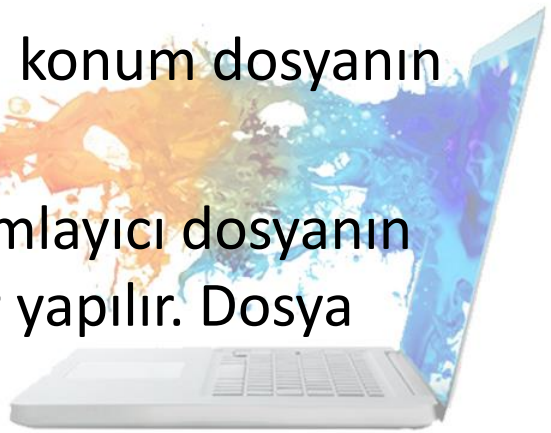
Resim 8.10: Tarayıcı ekran görüntüsü



Resim 8.11: Oluşturulan dosyanın içeriği

Dosya İşlemleri

- Resim 8.11'de görüldüğü gibi, dosya içeriğini sıfırlanmamış; var olan bilginin devamı olarak yeni bilgi dosyaya eklenmiştir. Bu durumun nedeni, Resim 8.9'da yer alan 4. satırda, "fopen" fonksiyonuna dosya adından sonra, 2. parametre olarak **"a+"** değerini vermemiş olmamızdır. Bu parametrenin alabileceği değerler ve anlamları aşağıdaki verilmiştir.
- **r**, dosya sadece okuma için açılır, açılma sonrası konum dosyanın başlangıcıdır
- **r+**, dosya hem okuma hem de yazma için açılır, konum dosyanın başlangıcıdır
- **w**, dosya sadece yazmak için açılır; dosya konumlayıcı dosyanın başlangıcına yerleştirilir ve dosya uzunluğu sıfır yapılır. Dosya mevcut değilse oluşturulmaya çalışılır.

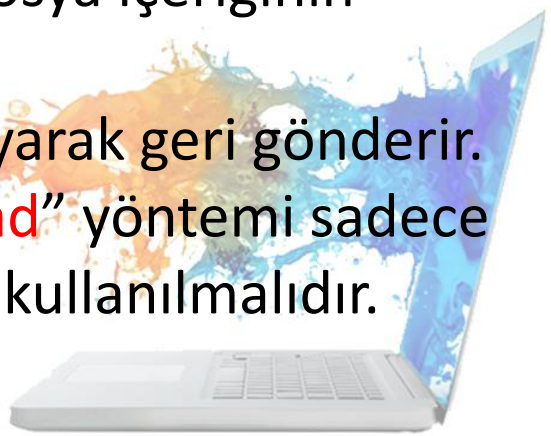


Dosya İşlemleri

- **w+**, dosya hem okumak hem de yazmak için açılır; dosya konumlayıcı dosyanın başlangıcına yerleştirilir ve dosya uzunluğu sıfır yapılır. Dosya mevcut değilse oluşturulmaya çalışılır.
- **a**, dosya sadece yazmak için açılır; dosya konumlayıcı dosyanın sonuna yerleşti rilir. Dosya mevcut değilse oluşturulmaya çalışılır.
- **a+**, dosya hem okumak hem de yazmak için açılır; dosya konumlayıcı dosyanın sonuna yerleştirilir. Dosya mevcut değilse oluşturulmaya çalışılır.
- **x** , dosya oluşturulur ve sadece yazmak için açılır; dosya konumlayıcı dosyanın başlangıcına yerleştirilir.
- **x+**, dosya oluşturulur ve hem okumak hem de yazmak için açılır; dosya konumlayıcı dosyanın başlangıcına yerleştirilir.

Dosya İşlemleri

- **Dosya İçeriğinin Okunması**
- PHP ile bir dosyanın içeriğinin yazılmasının yanı sıra, içeriğin okunması da gerekebilir. Bu tür durumlarda PHP dili bizlere yine 2 metot sunmaktadır.
- Bu metotlar aşağıdaki gibi sıralanabilir.
  1.file_get_contents
  2.fread
  Bu metotlardan ilki olan "file_get_contents", dosya içeriğinin tamamını bir metin değeri gibi geri gönderir.
  İkinci yöntem ise, dosya içeriğini satır satır okuyarak geri gönderir. Bu nedenle nadir bir kullanıma sahip olan "fread" yöntemi sadece çok büyük boyutlara sahip dosyalar okunurken kullanılmalıdır.

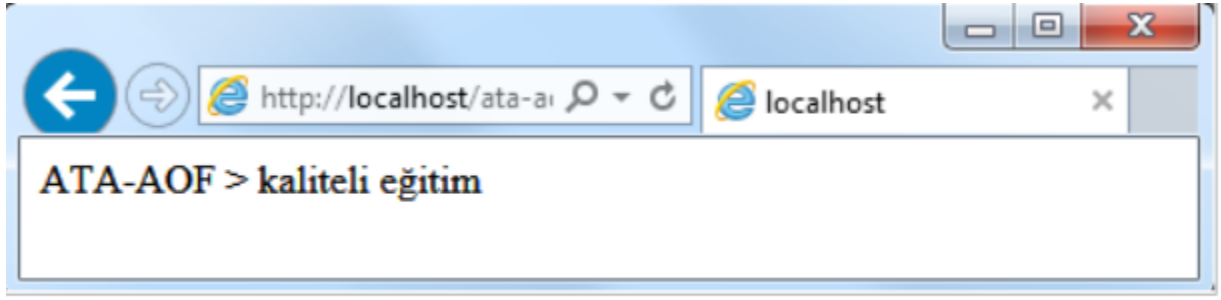


Dosya İşlemleri

- **Dosya İçeriğinin Okunması**
- Önceki konuda yazdığımız “aof.txt” dosyasının içeriğini alarak; tarayıcı ekranına basan bir PHP betiği ve ekran görüntüsü aşağıdaki gibi olabilir.

```php
<?php
$icerik = file_get_contents( "arsiv/aof.txt" );

echo $icerik;
?>
```

Resim 8.12: Dosya içeriğinin okunması için PHP kodları



Resim 8.13: Okunan içeriğin tarayıcıda oluşturduğu ekran görüntüsü

Dosya İşlemleri

- **Dosya İçeriğinin Okunması**
- Resim 8.12'de dosyanın içeriği okunduktan sonra elde edilen değer tamamen bir metin (string) şeklindedir.
- PHP dilinde yer alan metin işleme fonksiyonları ile elde edilen değişken üzerinde düzenleme işlemleri kolaylıkla yapılabilmektedir. "file_get_contents" fonksiyonu ile ayrıca bir internet adresi içeriği alınabilir.
- Bu durumda elde edilen veri, HTML etiketlerinden meydana gelen sitenin kaynak kodları olacaktır. Bu durumu anlamak için aşağıdaki PHP kodları ve ekran görüntüsü incelenebilir.

Dosya İşlemleri

- **Dosya İçeriğinin Okunması**

```php
<?php
$icerik = file_get_contents( "http://ataaof.edu.tr/" );
var_dump( $icerik );
?>
```

Resim 8.14: Bir internet sitesinin okutulması

```
string '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Stri
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="tr"
        <head>
                <title>Atatürk Üniversitesi Açıköğretim
```

Resim 8.15: Tarayıcı ekranında okunan içeriğin yazdırılması

Dosya İşlemleri

- **Varlık Kontrolleri**
- Bazı durumlarda dosya veya klasörün sunucuda var olduğunun kontrolü gerekebilir. Dosya ve klasörün fiziksel olarak sunucu sabit diskinde varlığını tespit için aşağıdaki fonksiyonlar kullanılabilir.
- **file_exists** – dosyanın varlık tespiti
- **is_dir** – dizinin varlık tespiti

Dosya İşlemleri

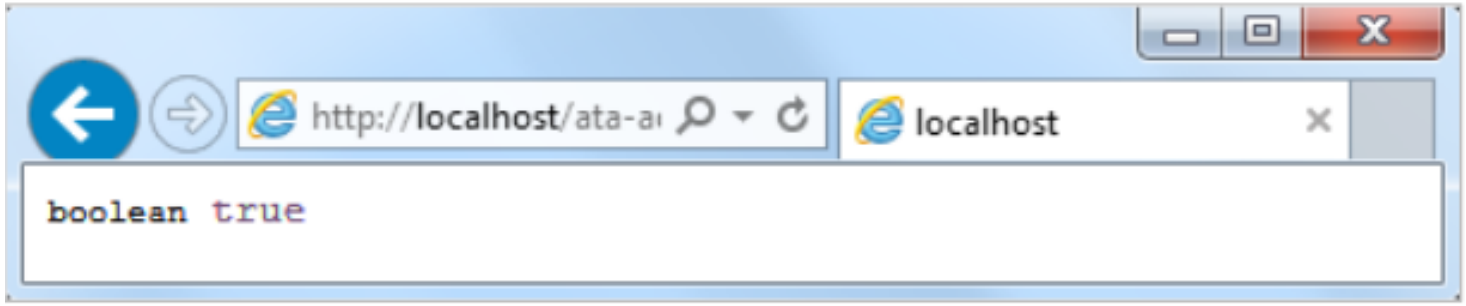
- **Dosya varlık kontrolü**
- Özellikle dosya işlemlerinde hata mesajlarının kaynağı genellikle dosyanın var olmamasından kaynaklanmaktadır.
- Bu tür olumsuz durumların önüne geçmek için PHP dilinde "**file_exists**" fonksiyonu kullanılmaktadır.
- Parametre olarak sadece dosyanın adını alan "file_exists" fonksiyonunun kullanımı sonrasında, "true" veya "false" gibi doğruluk (boolean) değeri geri dönecektir.
- Örnek olarak aşağıdaki kod parçası ve tarayıcıdaki ekran görüntüsü incelenebilir.

Dosya İşlemleri

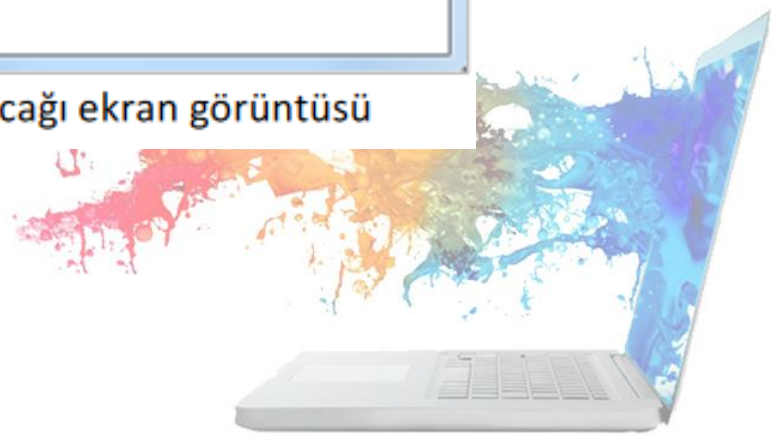
- **Dosya varlık kontrolü**

```php
<?php
$varlik = file_exists( "arsiv/aof.txt" );
var_dump( $varlik );
?>
```

Resim 8.16: Dosya varlığının PHP ile kontrolü için örnek betik



Resim 8.17: Varlık kontrolünün tarayıcıda oluşturacağı ekran görüntüsü



Dosya İşlemleri

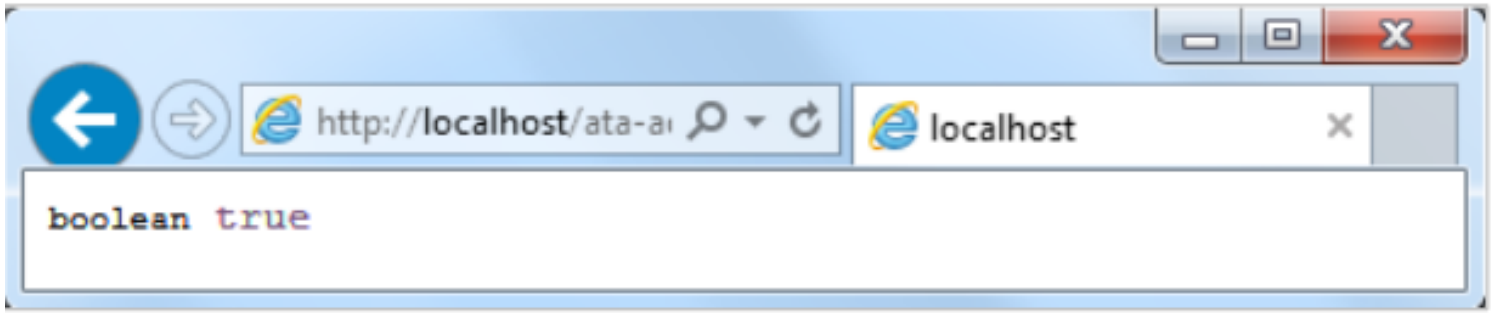
- **Dizin varlık kontrolü**
- Bir klasörün sunucuda var olduğunu kontrol etmek gerekebilir. Bu durumlarda "**is_dir**" fonksiyonu kullanılmaktadır.
- Fonksiyon, parametre olarak sadece dizinin yolunu beklemektedir. **"is_dir"** fonksiyonu çalıştırıldığında cevap olarak, **"file_exists"** fonksiyonu gibi, "true" veya "false" gibi doğruluk (boolean) değeri geri göndermektedir.
- Bu nedenle çıktıları tarayıcı ekranında görebilmek için "var_dump" fonksiyonu kullanmak gerekmektedir.
- Örnek PHP kodları ve tarayıcı görüntüsü aşağıdaki gibi verilebilir.

Dosya İşlemleri

- **Dizin varlık kontrolü**

```php
<?php
$varlik = is_dir( "arsiv" );
var_dump( $varlik );
?>
```

Resim 8.18: Dizinin varlık kontrolü için PHP kodları



Resim 8.19: Varlık kontrolü sonucunun tarayıcı ekranında görüntülenmesi

Dosya İşlemleri

- **Dizin varlık kontrolü**
- Dosya varlık kontrolü için kullanılan **"is_dir"** fonksiyonun bir benzeri olan **"is_file"** fonksiyonu da PHP dilinde mevcuttur.
- Bir dosyanın varlığı ayrıca bu fonksiyon ile de kontrol edilebilmektedir. Ancak; "is_file" fonksiyonu sadece sıradan dosyalar için "true" cevabını vermektedir.
- Örneğin; bir simge dosya (link) için "is_file" komutu, varlık mevcut olsa bile, "false" değerini geri gönderecektir.

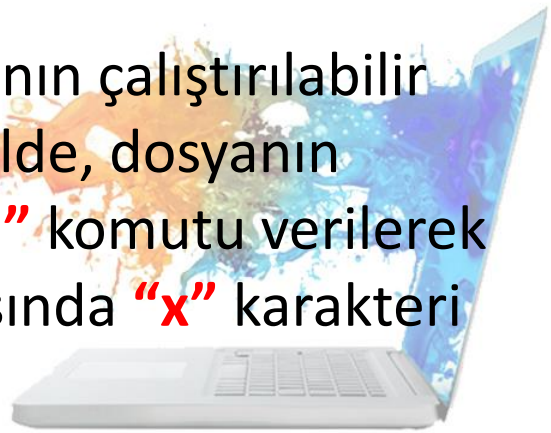
Örnek

•dizin_mi.php

```
<?php
function dizin_mi ( $a ) {
      print_r( is_dir($a) );
}

dizin_mi("C:\Windows");
?>
```

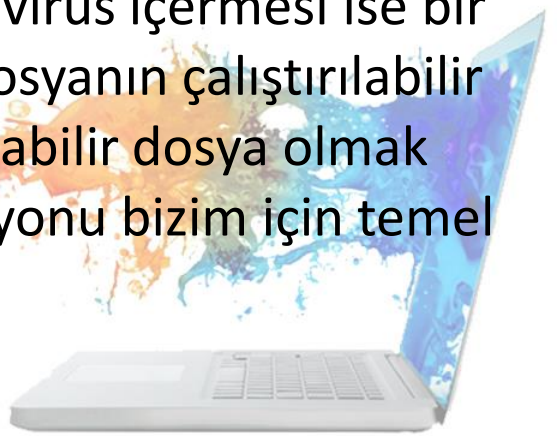
Dosya İşlemleri

- Ayrıca; bir dosyanın çalıştırılabilir olduğu da sorgulanabilir. Çalıştırılabilir dosyalar, Windows işletim sisteminde **"exe"** uzantısına sahiptir.
- PHP uygulama geliştirme dilinde bir dosyanın çalıştırılabilir olduğu, **"is_executable"** fonksiyonu kullanılarak tespit edilebilir.
- **is_executable("C:\Windows\cmd.exe");** gibi örnek bir kullanıma sahip olan fonksiyon cevap olarak "true" veya "false" geri gönderecektir.
- Linux ve MacOS işletim sistemlerinde bir dosyanın çalıştırılabilir yapıya sahip olduğunu görebilmek için terminalde, dosyanın bulunduğu klasöre geçiş yapıldıktan sonra **"ls -l"** komutu verilerek görülebilir. Çalıştırılabilir dosyaların izinleri arasında **"x"** karakteri mutlaka görülecektir.



Dosya İşlemleri

- PHP dilini kullanarak bir dosyanın çalıştırılabilir olduğunun sorgulandığı iki yaygın kullanım noktası mevcuttur. Bu noktalardan ilki; kullanıcıların yüklediği dosyaların sorgulanması; diğeri ise zamanlanmış görevlerde dosyanın çağırılması aşamasındadır.
- Geliştirdiğimiz internet sitelerinde bazı formlarda kullanıcılardan dosya yüklemesini talep edebiliriz. Örneğin; bir yardım talep formunda kullanıcılardan ekran görüntüsünü içeren resim istenmesi sık karşılaşılan bir durumdur.
- Kullanıcılar tarafından sisteme yüklenen dosyaların virüs içermesi ise bir güvenlik sorundur. İşte tam bu noktada yüklenen dosyanın çalıştırılabilir olması sorgulanabilir. Çünkü virüs dosyaları çalıştırılabilir dosya olmak zorundadır. Yani PHP dilindeki **is_executable** fonksiyonu bizim için temel seviyede bir virüs tarayıcı gibi çalışabilir.



Dosya İşlemleri

- Dosya işlemleri ile çalışılması halinde işletim sistemi üzerindeki boş disk alanının aralıklarla kontrol edilmesi büyük önem taşımaktadır. Özellikle ziyaretçi kayıtlarının (log) dosyalara kaydedilmesi genellikle bir müddet sonra disk alanının dolmasına sebebiyet vermektedir. Bu durumun ortaya çıkardığı asıl sorun ise disk kapasitesinin dolması sonucunda ortaya çıkan hataların genellikle birbiriyle alakasız olmasıdır.
- Örneğin; bir süredir sorunsuz olarak yayında olan internet siteniz birden hatalar vermeye başlayabilir. Bu nedenle belirli periyotlarla disk kapasitenizi kontrol etmenizde fayda vardır. **disk_free_space("C:")**veya **diskfreespace("C:")** fonksiyonlarını kullanarak (iki fonksiyon da aynıdır), boş disk alanınızı öğrenebilirsiniz.

Dosya İşlemleri

- PHP dilinde dosya işlemler için sıklıkla kullanılan fonksiyonlar ve temel işlevleri aşağıdaki gibi listelenebilir.
- Ayrıca; bu fonksiyon listesinin tamamına http://be2.php.net/manual/tr/ref.filesystem.php adresinden ulaşabilirsiniz.
- **chmod** — eğer PHP'nin yeterli izinleri varsa; dosya izinlerini değiştirebilirsiniz.
- **copy** — parametre olarak verilen dosyayı kopyalar.
- **delete, unlink, unset** — dosyayı silmenizi sağlar.
- **fileatime** — dosyanın son açıldığı zamanı döndürür.
- **fileperms** — dosyanın izinlerini döndürür.

Uygulama Ödevi

- Yerel sunucunuzda bir dosya oluşturarak; içerisine adınızı alt alta beş defa yazdırın.

Kaynaklar

- PHP Kılavuzu - http://php.net/manual/tr/index.php
- PHP Tutorials - https://www.w3schools.com/php/default.asp
- PHP Videolu Temel Dersler İçin: Youtube/Php Türkiye
- https://php-mysql.org/
- Atatürk Üniversitesi Açık Öğretim Fakültesi