T.C.
MİLLİ EĞİTİM BAKANLIĞI

MEGEP

(MESLEKÎ EĞİTİM VE ÖĞRETİM SİSTEMİNİN
GÜÇLENDİRİLMESİ PROJESİ)

BİLİŞİM TEKNOLOJİLERİ

INTERNET PROGRAMCILIĞI - 4

ANKARA-2008

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğrenme materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**482BK0097**
ALAN	**Bilişim Teknolojileri**
DAL/MESLEK	**Web Programcılığı**
MODÜLÜN ADI	**Internet Programcılığı 4**
MODÜLÜN TANIMI	Bu modül, gerekli ortam sağlandığında programlama komutları yardımıyla form ve nesne uygulamalarının hazırlanması konularının verildiği öğrenme materyalidir.
SÜRE	40/32
ÖN KOŞUL	Internet Programcılığı 3 modülünü başarmış olmak
YETERLİK	Programlama içinde form ve nesneler kullanmak
MODÜLÜN AMACI	**Genel Amaç** Gerekli ortam sağlandığında programlama komutları yardımıyla form ve nesne uygulamaları hazırlayabileceksiniz. **Amaçlar** ➢ Programlama içinde form kullanımını öğrenerek uygulamalar yapabileceksiniz. ➢ Nesne ve dosyalama işlemlerini gerçekleştirebileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam** Bilişim teknolojileri laboratuvarı, işletme ortamı **Donanım** Projeksiyon, bilgisayar. web programlama yazılımlarını çalıştırabilecek yeterlikte bilgisayar, bilgisayar laboratuvarı, internet bağlantısı
ÖLÇME VE DEĞERLENDİRME	➢ Her faaliyet sonrasında o faaliyetle ilgili değerlendirme soruları ile kendi kendinizi değerlendireceksiniz. ➢ Öğretmen modül sonunda size ölçme aracı (uygulama, soru-cevap) uygulayarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

Özellikle 2000'li yıllardan sonra dünyada ve ülkemizde pek bilinmeyen sunucu teknolojilerinde PHP Internet programlama dili, bir hayli yaygınlaşarak çoğu yazılımcıların bildiği bir programlama dili hâline geldi. Özellikle PHP 5 sürümü ile eksik gibi görülen nesne yönelimli programlama hâline gelen PHP, sorunsuz bir program hâline gelmiştir.

Bu modül ile en basit kayıt formundan en karmaşık, güvenli alışveriş sitelerine kadar etkileşimli web sitelerinin nasıl oluşturulacağını kavrayacaksınız. Önceki modüllerde öğrendiğiniz PHP konuları ile hazırlayacağınız web siteleri hâlâ statik bir sitedir. Çünkü fiziksel olarak siz güncellemediğiniz sürece aynı kalır. Kullanıcılarınızın siteyle anlamlı bir etkileşimi zordur.

PHP son derece verimli bir dildir. Fazla pahalı olmayan tek bir sunucu kullanarak her gün yüz binlerce ziyaretçiye hizmet verebilirsiniz.

PHP web üzerinde kullanım için tasarlanmış olduğundan web ile ilgili çoğu işlevi gerçekleştirebilmek için pek çok yerleşik fonksiyona sahiptir. Birkaç satır kodla GIF resmi oluşturabilir, web servislerine ve diğer ağ servislerine bağlanılabilir.

PHP gibi bir dille formları kullanmak, sitelerinizin dinamik olmasını sağlayacaktır. Siteleriniz özelleştirilebilir ve gerçek zamanlı bilgiler içerecektir.

ÖĞRENME FAALİYETİ–1

AMAÇ

Programlama içinde form kullanımını öğrenerek uygulamalar yapabileceksiniz.

ARAŞTIRMA

- PHP'de dış kaynaklardan (kullanıcıdan) bilgi alma yollarını araştırınız.
- Form oluşturmada GET ve POST metotları arasındaki farkları araştırınız.

Araştırma işlemleri için üyelik gerektiren web sitelerinin üyelik işlemlerinin, alışveriş sitelerinin nasıl hazırlandığını düşünerek böyle birkaç web sitesini inceleyiniz.

1. FORMLAR

Tüm programlama dillerinde yazılan programlar, bazı değişkenleri kullanıcıdan alarak bu değişkenler üzerinde mantıksal denetleme ve döngüler sonucunda kullanıcıya belli bir tepki verir. Hazırladığınız web sayfalarında çeşitli amaçlarla formlar oluşturmak gerekebilir. Bunlar anket formları, kayıt formları vb. formlar olabilir. Bu formlara girilen bilgiler e-mail adreslerine gönderilebileceği gibi Database (veri tabanı) veya kayıtların tutulacağı bölgelere de gönderilebilir. Bu konumda; formların sayfa üzerine nasıl yerleştirileceğini ve nasıl çalıştırılacağını göreceğiz. PHP, bir web yazılımı geliştirme dili olduğundan kullanıcıdan gelecek değişkenler üç şekilde alınabilir:

- Kullanıcı, tarayıcıdaki formu doldurarak PHP'ye Post ya da Get metoduyla gönderir.
- Kullanıcının bilgisayarında daha önceden depolanmış bir cookie'den değişken alınır.
- Kullanıcı, tarayıcının adres satırına değişken yazarak Query String yoluyla değişkeni gönderir.

Form ziyaretçinin sizin istediğiniz bilgilerin yanı sıra ziyaretçinin bilgisayarından web sunucusu bilgisayara, daha birçok bilgiyi de beraberinde getirir. web tasarımcısı ve web programcısı olarak bu bilgileri bilmeye daima ihtiyacınız vardır. Söz gelimi, ziyaretçinizin Browser türünü ve sürümünü belirleyerek onu uygun sayfaya yönlendirmekten ziyaretçiden istediğiniz bilgilerin sunucuya ulaştığında nerde hangi değişkende tutulduğuna kadar gerekli birçok bilgi sunucu çevre değişkenleri ve sunucu değişkenleri dediğimiz dizilerde bulunur. Bütün HTTP Server programları için ortak ve web programcısı için önemli değişkenler şunlardır:

HTTP_ENV_VARS HTTP : Sunucu programın çalışmakta olan PHP dosyası için oluşturduğu çevre değişkenlerinin yazılı olduğu dizi değişken. Bu değişkenin içinde şu unsurlar bulunur:

HOSTNAME: Sunucunun IP adresi

SHELL: Unix sisteminde kullanılan Shell programı

HOSTTYPE: Sunucunun adı ve türü

OSTYPE: Sunucunun işletim sistemi tipi

HOME: Çalışan programın kök dizini

PATH: Çalışan programın sunucudaki yolu

HTTP_SERVER_VARS: Sunucu programın çalışmakta olan PHP dosyasına sunduğu bazı bilgilerin bulunduğu dizi değişken. Bu değişkenin içinde şu unsurlar bulunur:

PHP_SELF: Çalışan PHP programının bulunduğu dizin ve adı

PATH_TRANSLATED: Çalışan PHP programının fiziksel yolu

HTTP_GET_VARS: Bir Form'dan GET metoduyla alınan bilgilerin anahtar=değer çiftleri olarak kaydedildiği dizi değişken

HTTP_POST_VARS: Bir Form'dan POST metoduyla alınan bilgilerin anahtar=değer çiftleri olarak kaydedildiği dizi değişken

HTTP_USER_AGENT: Ziyaretçinin bilgisayarında kurulu Internet Browser programı

QUERY_STRING: Form ile bilgi alırken GET metodunu kullandığımız takdirde Browser'ın göndereceği bilgilerin tutulduğu değişken

REMOTE_ADDR: Ziyaretçinin bilgisayarına ISS tarafından atanmış IP adresi

REQUEST_METHOD: Form ile gelen bilgilerin gönderildiği metot: GET veya POST

REQUEST_URI: O anda çalışmakta olan PHP dosyasının adı ve varsa bu ada eklenmiş Query_String

SCRIPT_FILENAME: O anda çalışmakta olan PHP programının dosya adı

SCRIPT_URI: O anda çalışmakta olan PHP programının tam URL adresi

SERVER_ADDR: Sunucunun IP adresi

SERVER_PROTOCOL: Sunucunun HTTP protokolünün sürümü

HTML formlarında bilgi almak çok basittir.

<form> </form>: Form içerisindeki nesnelerin kullanılması için **<form> </form>** tagları kullanılmalıdır.

```
<form method=”değer” action=”değer” name=”değer”></form>
```

Method: Formun hangi yöntemle karşı tarafa gönderileceğini belirler. İki değeri vardır. Bunlar GET ve POST'tur.

Action: Formun hangi adrese gönderileceğini belirler.

Name: Formun adını belirler.

1.1. GET

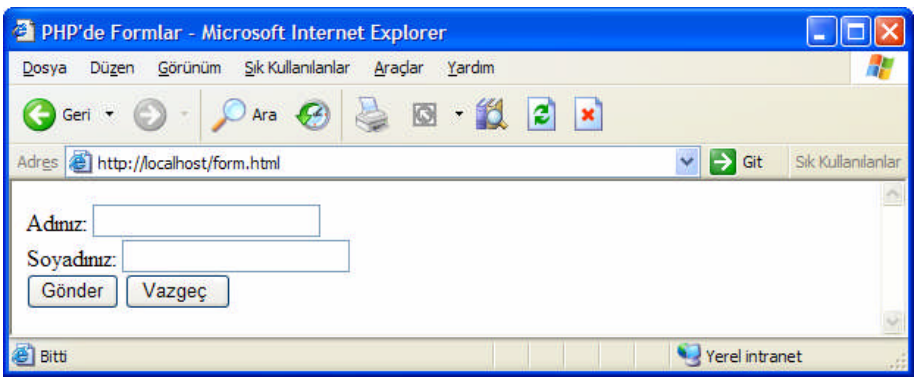
Forma girilen bilgilerin Action bölümünde belirtilen adrese veya sayfaya gönderilirken URL'nin arkasına eklenmesi biçiminde veri gönderimidir. Bu işlemle gönderilen bilgiler, tarayıcıların adres çubuğunda gösterilir. Bu yüzden güvenlik düşüktür. Özellikle şifre gönderimi gibi durumlarda bu metodun kullanılması pek sağlıklı değildir. Ayrıca, adres çubuğuna girilebilecek karakter sayısındaki kısıtlamalar nedeniyle fazla miktarda veri gönderimini engeller.

Form ile gelen bilgiler, GET metodu ile geliyorsa hem QUERY_STRING hem de HTTP_GET_VARS dizisine kaydolur. Şimdi GET metoduyla bir form dosyası oluşturalım. Basit bir HTML Form'u tasarlayalım. Aşağıdaki kodları form.htm adıyla kaydediniz:

```
<HTML>
<HEAD>
<TITLE>PHP'de Formlar</TITLE>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-
9">
<meta http-equiv="Content-Type" content="text/html; charset=windows-
1254">
</HEAD>
<BODY>
<FORM ACTION="form.php" METHOD="GET">
Adınız: <INPUT TYPE="TEXT" NAME="adi">
<br>
Soyadınız: <INPUT TYPE=TEXT" NAME="soyadi">
<br>
<INPUT TYPE="SUBMIT" VALUE="Gönder"> <INPUT TYPE="RESET"
VALUE="Vazgeç ">
</FORM>
</BODY>
</HTML>
```

Formun ACTION parametresine dikkat ederseniz, form.php adlı bir dosyanın adını göreceksiniz.

Bu, ziyaretçinin Gönder düğmesini tıklamasıyla birlikte Formun içerdiği bilgilerin METHOD parametresinde yazılı olan GET yöntemiyle Sunucuda gönderileceği programın adıdır. Bu sayfa, Browser'da şöyle bir görüntü verecektir: Şimdi bir an için ne olacağını düşünmeden, formu doldurun ve Gönder düğmesini tıklayın ve Browser'ınızdaki hata mesajına aldırmadan, URL adres kutusunda ne yazdığını okuyun.



Resim 1.1: form.html sayfası ekran görüntüsü

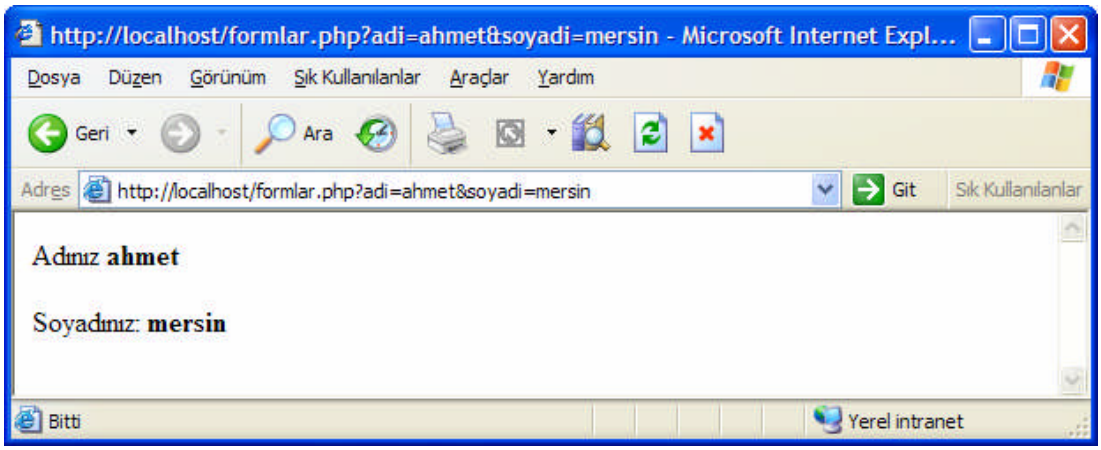
Yazılmış olan form.html dosyası browserda yukarıdaki resimdeki gibi olacaktır.

http://server/form.php?adi=ahmet&soyadi=mersin

Bu, HTTP protokolüne göre GET yoluyla bilgi göndermekte kullanılan yöntemin tam bir örneğidir. Browser, GET yoluyla bilgi göndereceği zaman, Form'daki bütün bilgileri URL-Encoding denen sistemle kodlar; Form'un alan adlarına o alanlara ziyaretçinin yazdığı bilgileri bir eşittir işaretiyle ekler; bu tür alan=girdi çiftlerinin arasına & (ve işareti) koyar ve gönderir. Web sunucu, bu bilgileri alınca önce kendi oluşturduğu bazı değişkenlere (hem QUERY_STRING, hem de HTTP_GET_VARS dizisine) yazar ve sonra URL hanesinde adı yazılı olan programa (sayfaya) verir. Şimdi bizim bu bilgilerin gönderildiği PHP programını kendisine verilecek bu bilgileri işlemeye hazır şekilde yazmamız gerekir. Şu aşağıdaki kodları form.php adıyla kaydediniz:

```
<?php
print ("Adınız <b>".$adi. "</b>\n\n");
print ("<p>Soyadınız:  <b>".$soyadi. " </b></p>\n\n");
?>
     veya
<?
$adi=$HTTP_GET_VARS["adi"];
$soyadi=$HTTP_GET_VARS["soyadi"];
Echo "Adınız : ". $ad. " <br> Soyadınız : ". $soyadi;
?>
```

Şimdi, browser'ınızda form.htm sayfasını yeniden açınız, Form'u doldurunuz ve gönderiniz. Açılacak sayfa, Form'un göndereceği bilgileri alacak ve kendi görüntüleyecektir.



Resim 1.2: form.php sayfası ekran görüntüsü

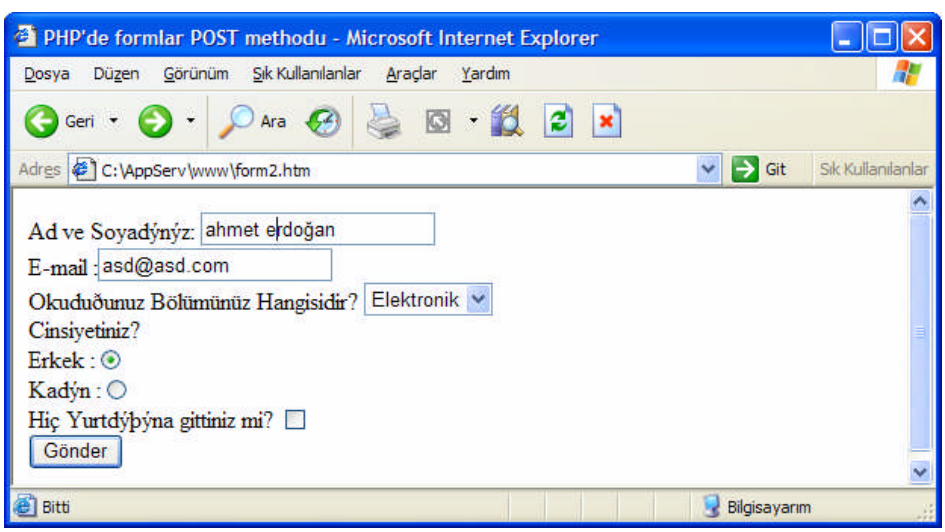
Yazılmış olan form.php dosyası browserda yukarıdaki resimdeki gibi olacaktır

Dikkat ederseniz, girilen bilgiler hem sayfada hem de adres çubuğunda görülmektedir. Fakat burada gördüğünüz gibi Sunucunun ziyaretçiden gelen bilgileri depoladığı dizileri kullanmadık. Bunu yaparken GET ile gelen bilgiler kendisinde verildiğinde PHP programının alan adlarını değişken adı, bunların karşısında yazılı olan verileri de bu değişkenin değeri saymasından yararlandık. Fakat istenilirse bu değişkenleri, Sunucunun oluşturduğu dizilerden de alabilirdik.

1.2. POST

Form'a girilen bilgilerin gönderilmesi için http sorgusunun kullanılması seçeneğidir. Gönderilen bilgiler açık olarak görülmez. Genel olarak en çok kullanılan seçenektir. Form'dan POST metoduyla gelen bilgiler HTML form etiketinin METHOD parametresinin değeri GET olabildiği gibi POST da olabilir ve HTTP sunucusu bu yöntemle gelen bilgileri $HTTP_POST_VARS dizi-değişkeninde tutar. GET ile POST metodun kullanımında hiçbir fark yoktur. Ama Internette kullanacağımız dosyaların güvenli olması açısından örneğin üyelik sisteminin bulunduğu bir sitede login olmak için girilen bir form alanına GET metoduyla yaparsanız, kötü niyetli kişiler bu linke tıklayarak siteye giriş yapabilir. Bu gibi durumlarda POST metodu kullanılmalı. Şimdi kapsamlı bir örnek vereceğiz. Alttaki örnek kodları form2.htm olarak herhangi bir editörde hazırlayınız (Frontpage, Dreamweaver, Not Defteri, Zend Studio gibi).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>PHP'de formlar POST methodu</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body>
<form action=" http://localhost/form2.php" method="post">
Ad ve Soyadınız: <input type="text" name="ad"><br>
E-mail :<input type="text" name="mail"><br>
Okuduðunuz Bölümünüz Hangisidir?
<select name="bolum">
<option value="bilgisayar">Bilgisayar
<option value="motor">Motor
<option value="elektrik">Elektrik
<option value="elektronik">Elektronik
</option>
</select><br>
Cinsiyetiniz?<br>
Erkek :<input type="radio" name="cinsiyet" value="erkek"><br>
Kadýn :<input type="radio" name="cinsiyet" value="kadin"><br>
Hiç Yurtdışına gittiniz mi? <input type="checkbox" name="yurt" value="yurt"><br>
<input type="submit" value="Gönder">
</form>
</body>
</html>
```

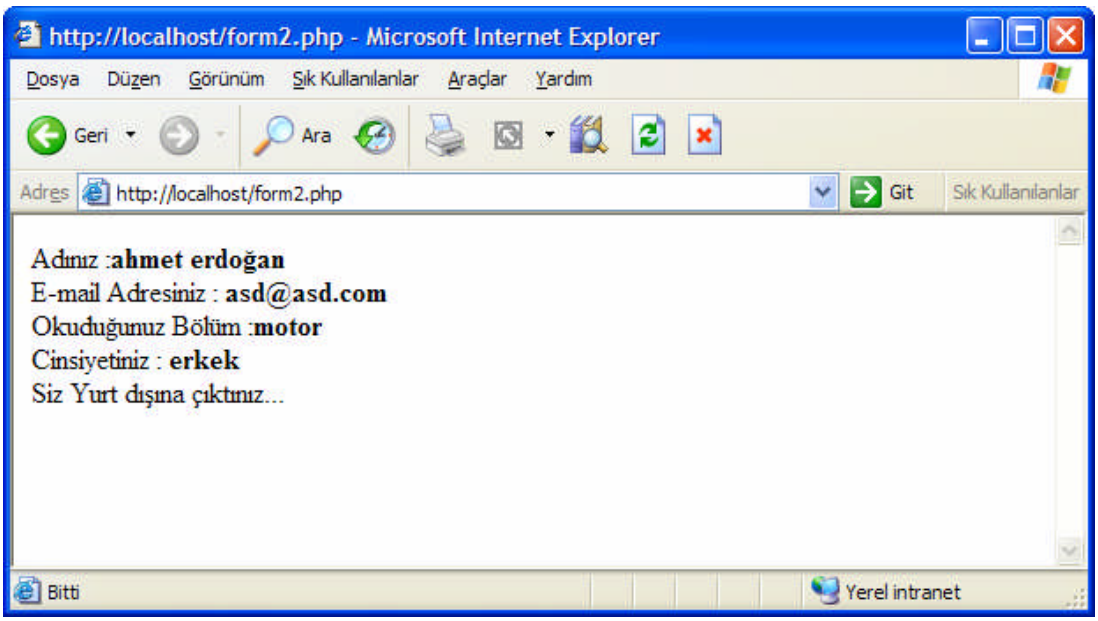


Resim 1.3: form2.htm sayfası ekran görüntüsü

Form.htm dosyasını tarayıcınızda açtığınızda yukarıdaki gibi bir ekran gelecektir.Bu formu doldurup gönder butonuna tıkladığınızda bilgilerin derlenip gönderileceği dosya olan form2.php dosyasının kodları ise şöyle olacaktır.

```
<?
echo  "Adınız :<b>$ad</b><br>";
echo "E-mail Adresiniz : <b> $mail</b><br>";
echo "Okuduğunuz Bölüm :<b>$bolum</b><br>";
echo "Cinsiyetiniz : <b>$cinsiyet</b><br>";
if (isset($yurt)){
  print "Siz Yurt dışına çıktınız...";
}
?>
```

Bu form alanı doldurulup gönder butonuna tıklandığında bilgiler POST metoduyla toplanıp form2.php dosyasına gönderilir. Form2.php dosyasındaysa gelen bu bilgileri yakalamak için değişken isimleri kullanıldı. Form alanı doldurulup gönder tıklandığında aşağıdaki gibi bir ekran oluşacaktır:



Resim 1.4: form2.php sayfası ekran görüntüsü

Bazı form alanında tanımlanan değişkenleri satır satır yazmak bazen sıkıntılara yol açar. Bu durumlarda PHP'nin $HTTP_GET_VARS ve $HTTP_POST_VARS dizi değişkenleri kullanılabilir. Bu dizi değişkenleri, formdan gelen bilgileri derleyip toparlayarak bir döngü yardımıyla tutulan değişken ve değerlerini ekrana basar. Şimdi yukarıda oluşturulan formdaki bilgileri ekrana yazdıralım. Yani form2.php aşağıdaki gibi yazalım:

```
<?
if (isset($HTTP_POST_VARS)){
  foreach ($HTTP_POST_VARS as $anahtar => $degeri){
          echo "$anahtar=$degeri <br>\n";
  }
}
if (isset($HTTP_GET_VARS)){
  foreach ($HTTP_GET_VARS as $anahtar => $degeri){
          echo "$anahtar =$degeri <br>\n";
  }
}
if (empty($HTTP_GET_VARS)||empty($HTTP_POST_VARS)){
  echo "herhangi bir bilgi gönderilmedi";
}
?>
```

Eğer form alanından POST metoduyla bilgi geldiyse ilk if deyimi devreye giriyor. Eğer form alanından GET metoduyla bilgi girildiyse bu sefer ikinci if deyimi devreye girer. Forma alanı direkt olarak form2.php dosyası çalıştırıldığında, en altta bulunan if deyimi çalışıp form alanının doldurulmadığını bildiren hata mesajı verecektir.

1.3. INPUT Tipleri

Form nesnelerinin birçoğu <form> tagının içerisinde kullanılan <input> taglarıyla oluşturulur.

```
<input type=”değer” name=”değer”>
```

Yukarıdaki Input tagında;
Name: Nesnenin adını belirler.
Type : Type parametresi, kullanılacak nesnenin tipini belirler. Bu tipler aşağıdaki gibidir.

1.3.1. Düz Metin Girişi (Plain Text Entry)

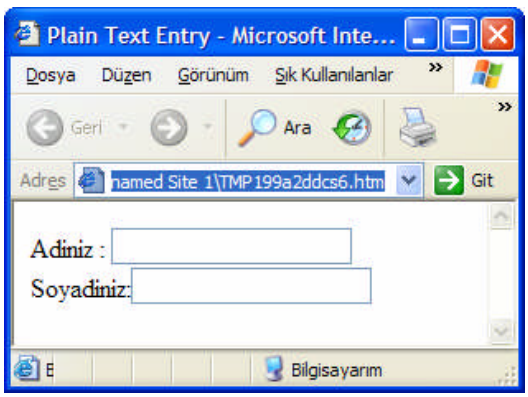
Düz metin kutusudur.

```
<input type="text" value="değer" maxlength="değer" size="değer">
```

Value: Metin kutusu içerisinde varsayılan olarak atanmasını sağlar.
Maxlength : Metin kutusu içerisine yazılabilecek karakter sayısını belirler.
Size : Metin kutusunun görüntüdeki uzunluğu belirler.

Örnek:

```
html>
<head>
<title>Plain Text Entry</title>
</head>
<body>
<form name="bilgiler"
method="post">
Adiniz : <input type="text"
name="ad"><br>
Soyadiniz:<input type="text"
name="soyad">
</form>
</body>
</html>
```



Resim 1.5: Düz metin girişi

1.3.2. Şifre Girişi (Password Entry)

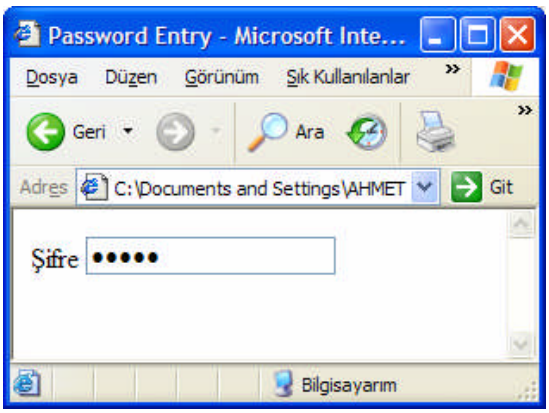
Metin kutusu görünümündedir, fakat bu alan içerisine bilgi girişi yapılırken yazılan karakterler asteriks "*" veya "●" karakterine dönüşür. Adından da anlaşılacağı gibi şifre alanlı formlarda kullanılır.

```
<input type="password"  maxlength="değer" size="değer">
```

Maxlength: Metin kutusu içerisine yazılabilecek karakter sayısını belirler.
Size: Metin kutusunun görüntüdeki uzunluğu belirler.

Örnek:

```
html>
<head>
<title>Plain Text
Entry</title>
</head>
<body>
<form name="sifre"
method="post">
Şifreniz: <input
type="password" name="sifre">
</form>
</body>
</html>
```



Resim 1.6: Şifre girişi

1.3.3. Yazı Alanı (Text Area)

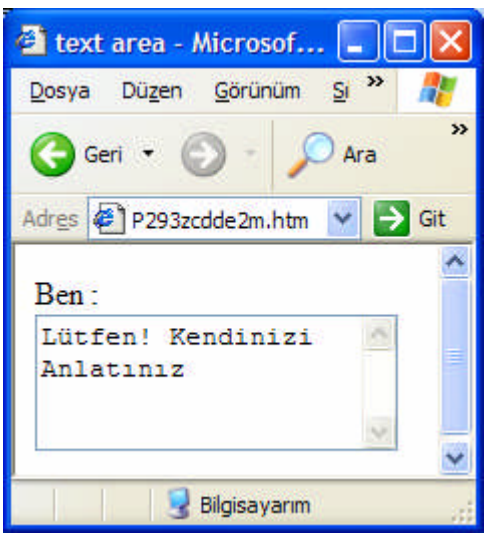
Bir açıklama ya da mail içeriği gibi uzun metinleri yazmak için kullanılır. Fakat diğerlerinde olduğu gibi <input> tagı ile yazılmaz. Onun yerine <textarea> tagı kullanılır.

```
<textarea rows=”değer” cols=”değer”>
```

Rows: Görüntülenecek karakter yüksekliğini belirler. Satır sayısını belirler.
Cols: Görüntülenecek karakter genişliğini belirler. Sütun sayısını belirler.

Örnek:

```
<html>
<head>
<title>text area</title>
</head>
<body>
<form name="bilgiler"
method="post">
  Ben :<br>
 <textarea rows=4 cols=20>
Lütfen! Kendinizi Anlatınız
 </textarea>
</form>
</body>
</html>
```



Resim 1.7: Yazı alanı

1.3.4. Seçim Kutusu (Select Box)

Visual Basic, Delphi vb. gibi görsel dillerdeki combobox (açılır kutu) nesnesine karşılık gelir. Kullanıcıya önceden belirlenmiş birkaç değerden bir tanesini seçtirmek için kullanılır.

<select> </select>: Seçim kutusunu oluşturmak için kullanılır. <select> tagı yalnız olarak kullanılmaz. <option> tagıyla beraber kullanılmalıdır.

<option>: <select> tagı içerisinde kullanılır. Menü elemanlarını belirlemek için kullanılır. </option> şeklinde kapatılmaz. Gerek yoktur.

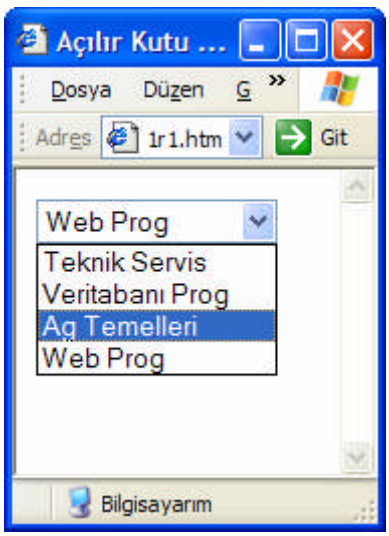
Selected: Eğer <option> tagında “selected” parametresi kullanılmışsa form ekrana görüntülendiğinde bu seçenek, seçili (varsayılan) olarak görüntülenmesi demektir.

Kullanımı:

```
<select>
        <option> seçenek1
        <option> seçenek2
        <option> seçenek3
        <option>  .....
        <option> seçenekN
        <option selected> seçenekK
</select>
```

Örnek 1:

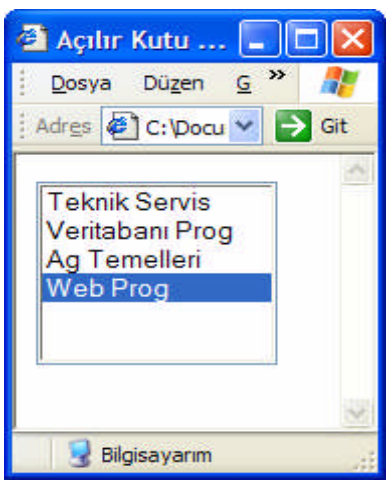
```
<html>
<head>
<title>Açýlýr
Kutu</title>
</head>
<body>
<form action="deneme" >
<select name="sec">
<option>Teknik Servis
<option>Veritabaný Prog
<option>Ag Temelleri
<option selected> Web
Prog
</select>
</form>
</body>
</html>
```



Resim 1.8: Seçim kutusu

Örnek 2:

```
</head>
<body>
<form action="deneme" >
<select name="sec"
multiple size="6">
<option>Teknik Servis
<option>Veritabaný Prog
<option>Ag Temelleri
<option selected> Web
Prog
</select>
</form>
</body>
</html>
```



Resim 1.9: Seçim kutusu

İkinci örnekte <select name=”sec” multiple> satırında “**multiple**” özelliğiyle birden fazla seçenek işaretlenebilir. Çoklu işaretleme yapabilmek için Shift veya Ctrl tuşlarıyla beraber Mouse sol tuşu kullanabilirsiniz. “ **size=”değer”** ” ifadesiyle de seçenek kutusunun kaç satır olacağı ayarlanabilir.

1.3.5. Onay Kutusu (Check Box)

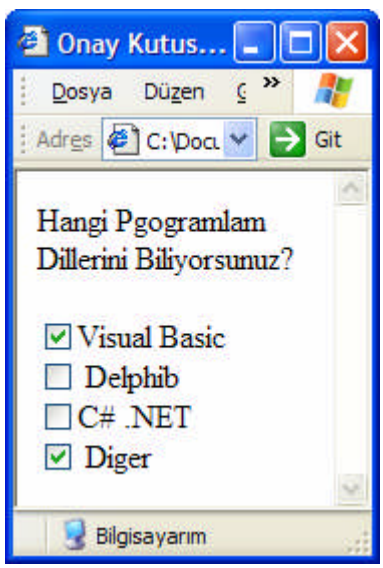
Check Box’lar, ziyaretçilerin birbirini takip eden birden fazla seçeneği aynı anda seçmelerine olanak sağlar.

<input type=”checkbox”> veya
<input type=”checkbox” checked> formatlarında kullanılır.

Checked: Form ekrana geldiğinde onay kutusunda işaretli olarak gelmesini sağlar.

Örnek :

```
<html>
<head>
<title>Onay Kutusu</title>
</head>
<body>
Hangi Pgogramlam Dillerini
Biliyorsunuz?<br>
<form action="deneme" >
<input type="checkbox" checked
name="onay">Visual Basic<br>
<input type="checkbox" name="onay1">
Delphib<br>
<input type="checkbox" name="onay2">C#
.NET <br>
<input type="checkbox" name="onay3"
checked> Diger
</form>
</body>
</html>
```



Resim 1.10: Onay kutusu

Örnek browserde açıldığında Visual Basic ve Diger seçeneklerinin varsayılan olarak seçili olduğu görülecektir. Bunun sebebi “checked” özelliğin verilmiş olmasıdır.

1.3.6. Seçenek Düğmesi (Radio Button)

Radio Button’lar, tek seçenekli durumlarda kullanılan bir tiptir. Seçeneklerden biri seçildiğinde diğerleri mutlaka seçili olmayacaktır. Sadece bir seçeneğin seçili olabilmesi için radio button’ların name özellikleri ,yani adları aynı olmalıdır.

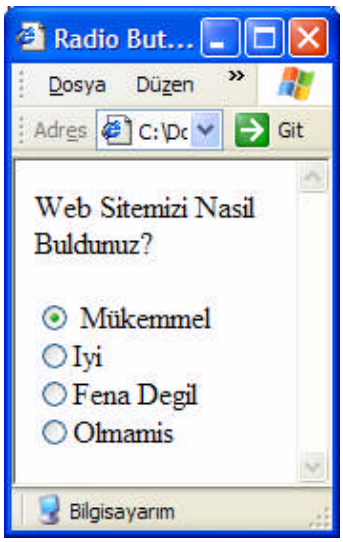
<input type=”radio”> veya

<input type=”radio” checked> formatlarında kullanılabilir.

Burada yine “checked” özelliği, form ekrana geldiğinde o seçeneğin işaretli gelmesini sağlar.

Örnek:

```
<html>
<head>
<title>Radio Button</title>
</head>
<body>
Web Sitemizi Nasil Buldunuz?
<form action="deneme" >
<input type="radio"  name="ona"checked>
Mükemmel <br>
<input type="radio" name="ona">Iyi<br>
<input type="radio" name="ona">Fena
Degil<br>
<input type="radio" name="ona">Olmamis
</form>
</body>
</html>
```



Resim 1.11: Seçenek düğmesi

1.3.7. Komut Düğmesi (Button)

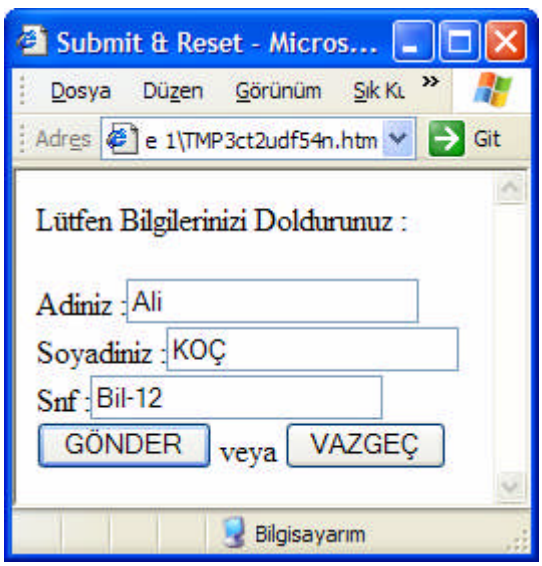
Formları hazırladıktan sonra formlara girilen bilgilerin ne yapılacağını belirlemek için buton eklemek gerekmektedir. Formlarda genellikle iki türlü buton vardır: Bunlardan birincisi “**Submit Buton**” dur. **Submit Buton**; Formlara girilen bilgilerin gönderilmesini sağlar. İkinci buton ise “**Reset Buton”** dur. **Reset Buton** ise Formlara girilen bilgilerin gönderilmemesi içindir, yani formdaki bilgilerin temizlenmesini sağlar.

```
<input type="submit" value="deger"> veya
<input type="reset" value="deger">
```

Buradaki;
Submit: Form bilgilerinin karşı tarafa gönderilmesini sağlar.
Reset: Form bilgilerinin karşı tarafa gönderilmeden temizlenmesini sağlar.
Value: Komut düğmesinin ekranda görüntülenecek etiketini belirler.

Örnek:

```
<html>
<head>
<title>Submit & Reset</title>
</head>
<body>
Lütfen Bilgilerinizi Doldurunuz :
<form action="deneme" >
Adiniz :<input type="text"  name="ona"><br>
Soyadiniz :<input type="text" name="ona1"><br>
Snf :<input type="text" name="ona2"><br>
<input type="submit" value="GÖNDER"> veya
<input type="reset" value="VAZGEÇ">
</form>
</body>
</html>
```



Resim 1.12: Komut düğmeleri

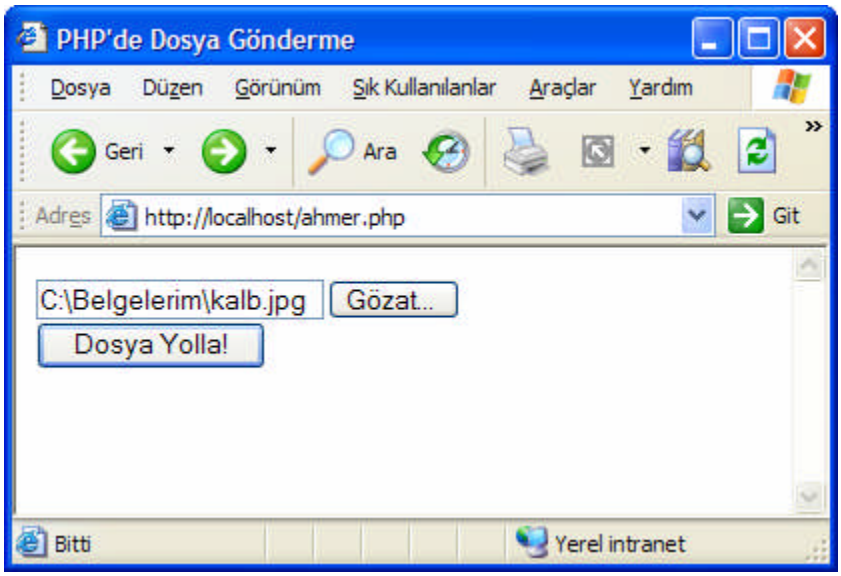
1.3.8. Dosya Gönderme (File Upload)

İnternetten hep dosya indiririz. Bir sunucuya, web ziyaretçisi olarak gönderebildiğimiz tek şey ise Formlara yazdığımız yazılardır. Oysa HTML'in INPUT etiketinin çok az kullanılan TYPE="file" parametresi, ziyaretçiye web sunucusuna dosya gönderme (upload) imkânı sağlar. HTTP protokolü buna imkân vermekle birlikte browser'lar bu imkânı kullanmaya ileri sürümlerinde kavuştular. PHP, ziyaretçilerimizin sitemize dosya göndermeleri hâlinde bu dosyaların yönetimine ayrıca kolaylık sağlayan değişkenlere sahiptir.

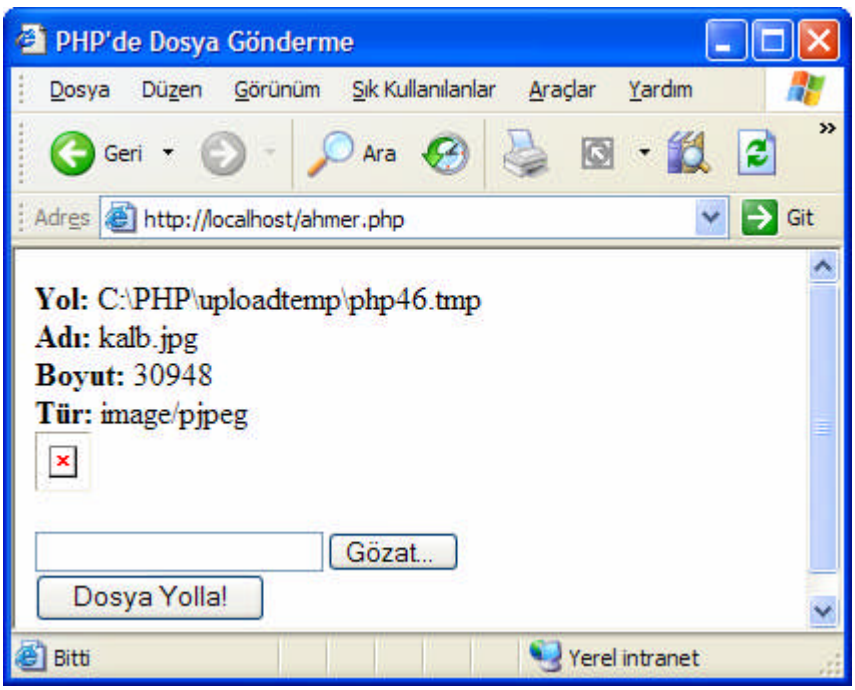
Aşağıdaki kodları dosyagonder.php olarak kaydediniz ve localhostta tarayıcınızdan deneyiniz.

Örnek:

```
<HTML>
<HEAD>
<TITLE>PHP'de Dosya Gönderme</TITLE>
<meta http-equiv=\"content-type\" content=\"text/html; charset=ISO-8859-9\">
<meta http-equiv=\"Content-Type\" content=\"text/html; charset=windows-1254\">
</HEAD>
<?php
$dosya_dizin = "";
$dosya_url = "http://localhost/";
if ( isset ( $dosya_gonder )) {
print ("<b>Yol:</b>".$dosya_gonder. "<br>\n");
print ("<b>Adı:</b>". $dosya_gonder_name. "<br>\n");
print ("<b>Boyut:</b>". $dosya_gonder_size. "<br>\n");
print ("<b>Tür:</b>". $dosya_gonder_type. "<br>\n");
copy ( $dosya_gonder, $dosya_dizi. "/".$dosya_gonder_name)or die ("Dosya kopyalanamıyor!");
if ( $dosya_gonder_type == "image/gif" ||$dosya_gonder_type == "image/pjpeg" ) {
print ("<img src=".$dosya_url."/".$dosya_url.">");  }
}
?>
</BODY>
<FORM ENCTYPE="multipart/form-data" ACTION="<?php print $PHP_SELF?>" METHOD="POST">
<INPUT TYPE="hidden" NAME="MAX_FILE_SIZE" VALUE="951200">
<INPUT TYPE="file" NAME="dosya_gonder"><BR>
<INPUT TYPE="SUBMIT" VALUE="Dosya Yolla!">
</FORM>
</BODY>
</HTML>
```



Resim 1.13: Dosya gönderme



Resim 1.14: Dosya gönderme

Bu programda <INPUT TYPE="file" NAME="dosya_gonder"> etiketinde kullandığımız NAME parametresine verdiğimiz değer, ziyaretçimizin göndereceği dosyanın sunucu tarafından kaydedileceği geçici dizinin tam yolunun yazılacağı değişkenin adı olacaktır. PHP, bu dosya ile ilgili her türlü bilgiyi bu adla kaydedecektir. PHP, ziyaretçiden bir dosya başarıyla aktarıldığı anda otomatik olarak bu isimden yararlanarak şu değişkenleri oluşturur:

$dosya_gonder: Geçici kayıt dizini yolu
$dosya_gonder_name : Ziyaretçinin gönderdiği dosyanın adı
$dosya_gonder_size : Ziyaretçinin gönderdiği dosyanın boyutu
$dosya_gonder_type: Ziyaretçinin gönderdiği dosyanın türü
PHP ayrıca bu bilgileri $HTTP_POST_FILES dizi-değişkeninde de tutar.

Yukarıdaki programda şu iki değişken çok önemlidir:

```
$dosya_dizin = "/";  // localhost içinde dosyanın kopyalanacağı klasör
$dosya_url = "http://localhost/";  // yerel intranette çalıştığımız için localhost yazıyoruz.
```

$dosya_dizin adıyla oluşturduğumuz değişkene vereceğimiz değer, ziyaretçinin göndereceği dosyanın kopyalanacağı klasörün adı olarak kullanılacaktır. Söz gelimi Windows ortamında buraya kişisel web sunucunun varsayılan klasörünün adını yazabilirsiniz. Ziyaretçinin göndereceği dosya, bir GIF biçiminde grafik dosyası ise bunu Browser'da görüntüleyeceğimiz için bu dizinin web'e açık olması; başka bir deyişle bizim web sunucumuzun erişebileceği bir dizin olması gerekir. Nitekim $dosya_url değişkenine değer olarak bu klasörün URL adresini veriyoruz. Bu iki değişkeni gerçek web sunucu için yazacağımız zaman bizim sunucumuzun varsaydığı fiziksel klasör adını ve yolunu bulmamız gerekir.

Bu uygulamayı kendi sunucunuzda yapmak isterseniz mutlaka bu iki değişkeni doğru yazmanız gerekir.

1.4. İçerik Kontrolü

Formlara girilen bilgileri kontrol ettirmek, bazı durumlarda gereklidir. Örneğin; e-mail adresinin zorunlu olması ve e-mail formatının doğru yazılmasını sağlamak istenebilir. Ziyaretçiler bu gibi durumlarda uyarılırlarsa daha doğru bilgiler girebileceklerdir. Bu durum güvenlik açısından çok önemlidir. Bazı durumlarda mesela kullanıcı kaydı yaptığınız bir formda kullanıcı adı ve şifre kısmının doldurulması mecburi olmalıdır. Ziyaretçiler, form alanlarında hiçbirini doldurmadan direkt gönder butonunu tıkladığında "Formu doldurduğunuz için teşekkür ederiz" gibi saçma bir yazı gelmesini hiç kimse istemez. Bunun için ziyaretçimizi uyarmamız gerekir.

Form alanlarını eksik doldurmayı engellemek if deyimi ile yapılacaktır. Tabi bu işi, birkaç yöntemle yapmak mümkündür. Aşağıda bu yöntemlerin de bazılarını inceleyeceğiz. Bunlardan uygun olanı kullanmak mümkündür. İçerik kontrolü yapmak için empty( ), isset( ) vb. php fonksiyonları kullanılabilir.

1.4.1. Isset ( )

Bu fonksiyon, aslında bir değişken içinde değer bulunup bulunmadığını sınar. Eğer değişkenin içinde değer bulunuyorsa **isset ()** doğru, yani **TRUE** sonucunu verecektir. Eğer

değişkenin içine bir değer atanmamış ise **FALSE** sonucunu verir. Değişken içerikli bir örnek verelim.

Örnek 1:

```
If (isset($deg) );
echo (“Değişkende değer var”);
else:
echo (“Değişkende değer yok”);
endif;
```

Örnek 2:

Bir formda kullanıcı adı ve soyadı girildiğini düşünelim. Bu alanlara bilgi girilmesi zorunludur. Bu alanların doldurulup doldurulmadığını kontrol eden php dosyası aşağıdaki gibi olabilir. Bu örnekte isset() fonksiyonu önünde “!” karakteri kullanarak isset() fonksiyonunun tam tersi olarak işlev görmesini sağlayacağız.

```
<?
if(!isset($ad)) {
Print “Ad Kısmını Doldurmadınız. Lütfen Doldurunuz.”;
}
Elseif (!isset($soyad)){
Print “Soyad Kısmını Doldurmadınız. Lütfen Doldurunuz.”;
}
else {
Print “Adınız : <b> “.$ad.” </b> <br>Soyadınız :<b>”.$soyad.”</b>’dir”;
}
?>
```

1.4.2. Empty ( )

Bu fonksiyon da isset() fonksiyonuyla aynı işlevi görür. Ama sorgulaması isset() fonksiyonunun tam tersidir. Bu fonksiyon, değişkenin içerisine eğer değer yoksa **TRUE** değerini verir. Aksi takdirde **FALSE**

Örnek 1: Bir değişkende değer olup olmadığının kontrolü.

```
if (empty($deg)){
print “Değişken değeri boş”;
}
Else{
Print “değişkende değer var”;
}
```

NOT: isset() fonksiyonu ve empty() fonksiyonunu dilediğiniz şekilde kullanabilirsiniz. Mesela isset() fonksiyonu önüne “!” ünlem işareti koyarak empty

fonksiyonu gibi kullandık. Yine empty() komutu önüne "!" ünlem işareti koyarak isset() komutu gibi kullanılabilir.

Örnek 2: Bir formda kullanıcı adı ve soyadı girildiğini düşünelim. Bu alanlara bilgi girilmesi zorunludur. Bu alanların doldurulup doldurulmadığını kontrol eden php dosyası aşağıdaki gibi olabilir:

```
<?
if(empty($ad)) {
Print ”Ad Kısmını Doldurmadınız. Lütfen Doldurunuz.”;
}
Elseif (empty($soyad)){
Print “Soyad Kısmını Doldurmadınız. Lütfen Doldurunuz.”;
}
else {
Print “Adınız : <b> “.$ad.” </b> <br>Soyadınız :<b>”.$soyad.”</b>’dir”;
}
?>
```

1.4.3. If ( ) Komutu

Doğrudan if deyimi ile değişkenlerde değer olup olmadığını veya girilen değerlerin doğru olup olmadığının kontrol edebiliriz. Burada doğrudan değişkenin içini boş yapıyoruz. Eğer formda veya değişkende boş alanlar varsa if devreye girip kontrol eder. Değilse program normla seyrini devam ettirir.

Örnek 1: Bir formda kullanıcı adı ve soyadı girildiğini düşünelim. Bu alanlara bilgi girilmesi zorunludur. Bu alanların doldurulup doldurulmadığını kontrol eden php dosyası aşağıdaki gibi olabilir:

```
If ($ad= =””){
Print ”Ad Kısmını Doldurmadınız. Lütfen Doldurunuz.”;
}
Elseif ($soyad = = “” ){
Print “Soyad Kısmını Doldurmadınız. Lütfen Doldurunuz.”;
}
else {
Print “Adınız : <b> “.$ad.” </b> <br>Soyadınız :<b>”.$soyad.”</b>’dir”;
}
?>
```

Örnek 2: Kullanıcı adı ve şifre kontrolü yapan bir programdır.

Sifre.php

```
<?
If ($kullanici= =”” and $sifre= =””){
Print” kullanıcı adını ve ya şifre girmediniz”;
}
```

```
Elseif ($kullanici= =”mugla” and $sifre= =”4833”){
Print “Kullanıcı adınız mugla ve şifreniz 4833”;
}
Elseif ($kullanici= =”bilgisayar” and $sifre= =”bil11”){
Print “Kullanıcı adınız bilgisayar ve şifreniz bil11”;
}
Else{
Print “Kullanıcı adınız veya şifreniz yanlış”;
}
?>
```

Sifre.html

```
<form action=”sifre.php” method=”post”>
Kullanıcı Adı:<input type=”text” name=”kullanici”><br>
Şifre: :<input type=”text” name=”sifre”><br>
<input type=”submit”  value=”Giriş”>
</form>
```

1.4.4. Strlen( ) Fonksiyonu ile Karakter Sayısı Kontrolü

Formlarda güvenlik açısından girilecek değerin karakter sayısını sınırlamak da gerekebilir. Burada Strlen () fonksiyonu, verilen metnin uzunluğunu tam sayı cinsinden döndürür ve buna göre kontrol yapılabilir.

Örnek 1:

```
<?
Echo strlen(“Muğla”); // 5 değerini döndürür
?>
```

Yukarıdaki örneği çalıştırdığınızda ekranda 5 yazacaktır. “Muğla” text bilgisi 5 karakterden oluşmaktadır.

Örnek 2: Sitenize kullanıcı kaydı yapıyorsunuz ve şifrenin en 8 karakterden oluşmasını istiyorsunuz.

```
<?
If (strlen($sifre)<8){
Print “Şifreniz en az 8 karakterden oluşmalıdır”;
}
Else{
//devam edilecek komutları buraya yazınız
}
?>
```

Burada strlen() komutu sifre değişkenine girilen değeri karakter olarak sayar. Sifre değişkeni içerisine en az 8 karakter girilmesini sağlar. Aksi durumda hata mesajı verecektir.

1.4.5. Eregi() Fonksiyonu ile Formlarda E-mail Adresi Yazım Kontrolü

Metin içinde verilen kelimeyi, karakteri bulmak için kullanılır. Aranılan karakter sıralanışı bulunduğu takdirde doğru (true), bulamadığı takdirde yanlış (false) karşılığı bir değer verir. Bu fonksiyon büyük-küçük harfe duyarlıdır.

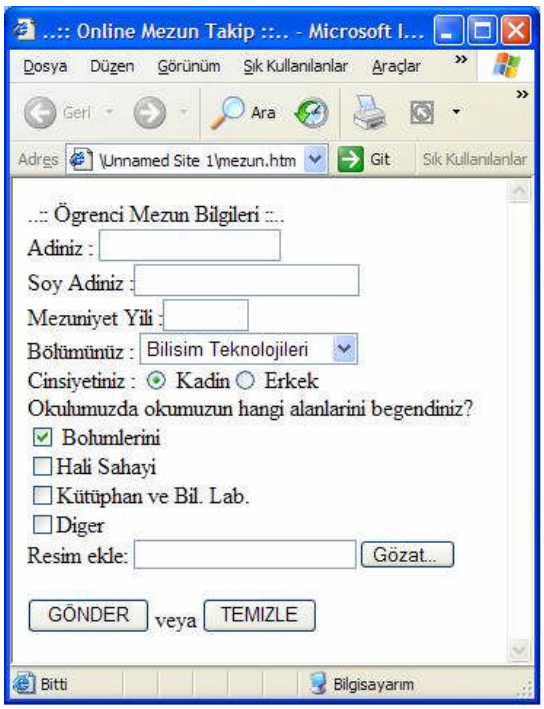
Örnek 1:

```
<?
  $aranan_ifade="Türkçe";
  $ifade="En Yararlı Türkçe Kaynak";

  if(ereg($aranan_ifade,$ifade)) {
    echo "aranan ifade bulundu"; }
  else
  {
    echo "aranan ifade bulunamadı"; }

 //ekran çıktısı: aranan ifade bulundu
?>
```

Örnek 2: Formda girilen e-mail adresinin doğru yazılıp yazılmadığının kontrolünü yapan php kodları.

```
<?php
 function eposta_kontrol($eposta) {
   if (!eregi ("^([a-z0-9_]|\\-|\\.)+@(([a-z0-9_]|\\-)+\\.)+[a-
z]{2,4}$", $eposta))
   die ("E-Posta Adresiniz Geçersiz");
}
?>
```

UYGULAMA FAALİYETİ



Resim 1.15: Online mezun takip

İşlem Basamakları	Öneriler
➢ Yukarıdaki resimdeki formu doldurup bilgileri gönderilecek bir form hazırlayınız. Bir mezun.htm adlı dosyada oluşturunuz.	
➢ Öncelikle form tagını oluşturunuz.	➢ Form metodunu post tavsiye ederiz ve yazacağınız diğer input typelarını bu form tagı içine yazınız.
➢ Adınızı, soyadınız, mezuniyet yılı input typelarını “text” olarak seçiniz.	
➢ Bölümünüz select box’ına en az 3 bölüm giriniz ve “Bilişim Teknolojileri” seçili olsun.	
➢ Okulun beğenilen seçeneklerinde “Bölümleri” seçili olsun.	
➢ Mezun öğrenci forma resim gönderebilsin.	
➢ mezun.php doyasını oluşturunuz ve bu girilen bilgileri ekranda yazdırınız.	➢ mezun.htm dosyasından gelecek değişken adlarına dikkat ediniz.
➢ Formdan gelen mezuniyet yılına göre kişinin okuldan mezun olalı kaç yıl olduğunu hesaplatınız.	➢ Tarih fonksiyonlarını kullanabilirsiniz.

ÖLÇME VE DEĞERLENDİRME

Bu bölümde yer alan ölçme sorularını dikkatlice okuyarak cevaplandırınız.

ÖLÇME SORULARI

Soruları okuyarak doğru bulduğunuz seçeneği işaretleyiniz.

1. web ortamında kullanıcıdan bilgi almak için formlarda aşağıdaki metotlardan hangisi kullanılır?
A) Put B) Get C) OutPut D) Send

2. Form bilgileri hangi tag komutları kullanılır?
A) <form> … </form> B) <form> … </script>
C) <script> … </form> D) Hiçbiri

3. Form metotlarından hangisi daha güvenlidir?
B) Get B)Submit C) Text D) Post

4. Aşağıdakilerden hangisi input type'ından **değildi**r?
A)Text B) Submit C) Post D) Password

5. Formlarda düz metin girişi için hangi input type özelliği kullanılır?
A)Text B) Password C) Select Box D) Check Box

6. Formlarda tek seçenekli durumlarda kullanılan input type'ı hangisidir?
A)Option B) Check Box C) Select Box D) Radio

7. Formları hazırladıktan sonra formlara girilen bilgilerin başka bir dosyaya gönderilmesi için hangi input type özelliği kullanılır?
A)Submit B) Reset C) Check Box D) Value

8. Bir değişken içinde değer bulunup bulunmadığını sınayan PHP fonksiyonu aşağıdakilerden hangisidir?
A)empty ( ) B) strlen ( ) C) isset ( ) D) eregi ( )

9. Formlarda girilecek değerin karakter sayısını sınırlamak için hangi PHP fonksiyonu kullanılır?
C) empty ( ) B) strlen ( ) C) isset ( ) D) eregi ( )

10. Değişkenim içerisi boş olduğunda (değer girilmemişse) TRUE değeri gönderen PHP fonksiyonu aşağıdakilerden hangisidir?
A)empty ( ) B) strlen ( ) C) isset ( ) D) eregi ( )

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Test içinde cevaplandıramadığınız, yanlış cevaplandırdığınız veya kendinizi bilgi bakımından eksik hissettiğiniz sorular için ilgili konulara tekrar dönünüz.

ÖĞRENME FAALİYETİ–2

AMAÇ

Nesne ve dosyalama işlemlerini gerçekleştirebileceksiniz.

ARAŞTIRMA

- Programlamada nesne kavramını araştırınız.
- Internette dosya oluşturma ve dosyalama işlemleri hakkında bilgi toplayınız.
- Internette, kullanıcılardan gelen bilgiler nereye, nasıl kaydediliyor? Araştırınız.

2. NESNELER

2.1. Nesne Oluşturma

Programlama dilleri, işlevlerine ve kullanım amaçlarına göre farklı yazılım geliştirme yöntemleri kullanır. Günümüzde birçok farklı programlama dili mevcuttur. Programlama dilleri, tarih içinde farklı ihtiyaçlara göre gelişim göstererek iki farklı yönde iki farklı disipline yönlenmiştir. Bunlar, tarih olarak eski olan ve günümüzde de hâlâ kullanılan fonksiyon tabanlı programlama dilleri ve daha yeni olan nesne tabanlı yazılım geliştirmeye izin veren programlama dilleridir. Yeni geliştirilen dillerin tümü, her iki disiplinde de yazılım geliştirmeye izin vermektedir.

Nesne yönelimli (Object-oriented) programlama icat edildiğinde geleneksel programlamaya aşina olanlar önce tereddüt ettiler; sonra bunun ne harika bir teknik olduğunu gördüler ve nesnelerden vazgeçemez oldular. Kabaca tanımlarsak nesne, kendi değişkenleri ve icra edeceği komutlardan oluşan fonksiyonları ile bir bütündür. Nesneyi bir kere tanımladıktan sonra istediğimiz kadar örneğini oluşturabiliriz. Bir nesnenin yapacağı işten, o nesnenin metodu diye söz ederiz. Bu açıdan bakarsanız programlarımızda nesneler sadece metotları için işe yarar. Bir nesne, aslında bilgisayarın hafızasında ayrılmış bir alandan başka bir şey değildir. PHP yorumlayıcısı, web sunucusunda okuduğu her satırın karşılığı olarak görev yaparak dosya sonuna geldiğinde de görevi bitirmek üzere tasarlanmış bir yorumlayıcıdır. Nesne tabanlı yazılmış PHP kodunda süreç şu şekilde gelişir: Dosyanın okunması, nesnelerin oluşturulması, nesnelerin çeşitli özelliklerinin değiştirilmesi, nesnenin ekrana gönderilmesi, işlemin bitirilmesi, nesnenin yok edilmesi. Bu süreçte yer alan nesne oluşturma ve nesnenin çeşitli özelliklerinin değiştirilmesi adımları, tamamen bellekte gerçekleşen işlemlerdir.

Nesne yönelimli yazılımlar, ihtiyaçlarımızı karşılamak üzere birbiriyle etkileşim içinde olan nitelikleri ve işlemleri olan bir nesneler kümesi olarak tasarlanır ve oluşturulur.

Nitelikler, nesneyle ilgili özellik veya değişkenlerdir. İşlemler ise nesnenin kendini değiştirmek veya haricî bir etki sağlamak amacıyla gerçekleştirebileceği metot, eylem veya fonksiyonlardır.

Bir web projesi, hyperlink (köprü) içeren sayfalardan karmaşık uygulamalara doğru giden bir evrim izler. İletişim kutuları, pencereler veya dinamik oluşturulmuş HTML sayfaları yoluyla temsil edilebilen karmaşık uygulamalar, geliştirme metodolojisi üzerinde ayrıntılı bir şekilde düşünülmeyi gerektirir. Nesne yönelimli yaklaşım ise size projelerinizdeki karmaşıklıkla başa çıkabilme, kodların yeniden kullanılabilirliğini artırma ve böylece kontrol maliyetini en aza indirebilme imkânını sunar.

Nesne yönelimli yazılımlarda nesne, belirli bir veri üzerinde çalışan depolanmış veri ve işlemlerin benzersiz ve tanımlanabilir bir toplamıdır.

Nesneler, sınıflar hâlinde gruplanabilir. Sınıflar (class) tek tek birbirinden farklı olabilen, ama bazı ortak noktaları olan bir nesneler kümesini temsil eder. Bir sınıf tümü aynı şekilde davranan birbirinin aynı işlemlere ve aynı şeyleri temsil eden birbirinin aynı niteliklere sahip nesneler içerir. Ancak bu niteliklerin değerleri, nesneden nesneye değişebilir.

2.1.1. Sınıf ve Nesne Tanımlama

Her nesne aslında bir türün örneğidir. Nesneler, türlerin özelliklerinin değiştirilmesinden oluşur. Türler, sadece nesnelerin genel özelliklerinden bahseder. Nesneler, türden alınan birer örnektir, genel türün özelliklerini taşımakla birlikte bazı şeylerde farklılıklar gösterebilirler. Tablo türünden bir örnek nesne hazırlandığında zemin rengi, satır-sütun sayıları, başlık ve açıklama bilgileri farklı olabilir. PHP' de en küçük sınıf tanımı, aşağıdaki gibidir:

```
class tablo1
{
}
```

Kullanışlı olabilmeleri için sınıfların niteliklere ve işlemlere ihtiyacı vardır. **Var** anahtar sözcüğünü kullanarak bir sınıf tanımı içinde değişkenler tanımlanabilir. Aşağıdaki kodlarda degisken1 degisken2 diye iki niteliği olan, tablo1 diye bir sınıf oluşturulmaktadır.

```
class tablo1
{
var $degisken1;
var $degisken2;
}
```

İşlemleri sınıf tanımı içinde fonksiyonlar tanımlayarak oluşturuyoruz. Aşağıdaki kod, hiçbir şey yapmayan iki işlemi olan tablo1 adlı bir sınıf oluşturur. islem1( ) fonksiyonu hiç parametre almazken islem2( ), iki tane alır.

```
class tablo1
{
Function islem1()
{
}
Function islem2($param1,$param2)
{
}
}
```

2.1.2. Sınıfların Örneklerini Oluşturmak

Bir sınıf tanımlandıktan sonra, üzerinde çalışabilmek için bir nesne oluşturmak gerekir. Bu bir sınıfın örneğini oluşturmak olarak da bilinir. Bir nesne **new** anahtar sözcüğü kullanılarak oluşturulur. Bunu yaptığınızda nesnenizin hangi sınıfın örneği olacağını belirlemeniz ve yapılandırıcının gerektirdiği parametreleri sağlamanız gerekir.

Aşağıdaki kod, bir yapılandırıcıyla tablo1 adlı bir sınıf tanımlar ve sonra da tablo1 tipinde üç nesne oluşturur.

```
<?php class tablo1
{
Function construct($param)
{
echo "Yapılandırıcı $param parametreyle çağrıldı";
}
}
$a=new tablo1;
$a->construct("Birinci");
echo"<br>";
$b=new tablo1;
$b->construct("İkinci");
echo"<br>";
$c=new tablo1;
$c->construct("");

?>
```

Yapılandırıcı, nesne oluşturduğunuz her seferde çağrıldığı ve construct isimli bir fonksiyonu çalıştırdığı için bu kod aşağıdaki çıktıyı verir:

```
Yapılandırıcı Birinci parametreyle çağrıldı
Yapılandırıcı İkinci parametreyle çağrıldı
Yapılandırıcı parametreyle çağrıldı
```

2.1.3. Sınıf Niteliklerini Kullanmak

Sınıf içinde, **$this** adındaki özel bir işaretçiye erişebilirsiniz. Eğer geçerli sınıfınızın bir niteliği **$degisken** adını taşıyorsa değişkene sınıf içindeki bir işlemden erişirken veya bu değişkeni ayarlarken **$this - > degisken** ifadesiyle kendine başvurabilirsiniz.

Aşağıdaki kodda bir sınıf içinde bir niteliği erişilmekte ve bir değer verilmektedir:

```
class sinif
{
Var $degisken;
Function islem($param)
{
  $this->degisken=$param;
  echo $this->degisken;
}
}
<HTML>
<HEAD>
<TITLE>PHP de Degiskenler</TITLE>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-
9">
</HEAD>
<BODY>
<B>
<H2>
<?php
class ogrenci {
// özellikleri tanımlayalım
var $adi;
var $soyadi;
var $sinav1;
var $sinav2;
var $not;
// metodları tanımlayalım
function adi_belirle ($n) {
$this->adi = $n;
}
function soyadi_belirle ($n) {
$this->soyadi = $n;
}
function sinav1_belirle ($n) {
$this->sinav1 = $n;
}
function sinav2_belirle ($n) {
$this->sinav2 = $n;
}
function not_hesapla() {
$this->not = ($this->sinav1 + $this->sinav2)/2;
```

```
print ($this->adi. " " . $this->soyadi . " için not ortalaması: ". $this->not);
}
}

//Buraya başka kodlar girecek
$ogr1 = new ogrenci();
$ogr1 -> adi_belirle("Zinde");
$ogr1 -> soyadi_belirle("Tombak");
$ogr1 -> sinav1_belirle(7);
$ogr1 -> sinav2_belirle(10);
$ogr1 -> not_hesapla();
?>
</H2>
</B>
</BODY>
</HTML>
```

Bu programda ogrenci adlı bir nesne tanımlıyoruz ve daha sonra bir değişken adına new komutu ile bu nesnenin bir örneğini oluşturuyoruz. Nesnelerin yeni bir örneği veya kopyasını çıkartmak ifadeleri aslında yaptığımızı tam anlatmıyor. Nesnenin tanımı bir adet; her new komutu ile bu nesnenin özelliklerine ve metotlarına sahip yeni bir nesne yapmış oluyoruz. Nitekim bu işe insanın dilini dolayan İngilizce bir kelimeyle Instantiation (yeni bir varlığını oluşturma) deniyor. Bu örnekte, $ogr1 adlı değişken gerçekte, ogrenci nesnesinin tam bir örneği: içinde beş değişken ve altı metot var. Nesneyi bir kere tanımladıktan sonra programın daha ileri aşamalarında bu istediğimiz kadar örneğini farklı isimler vererek oluşturabiliriz. Şimdi şu satıra dikkat edelim:

```
$ogr1 = new ogrenci();
$ogr1 -> adi_belirle("zinde");
```

Burada $ogr1`in parametrelerine nasıl değer yazdırdığımızı görüyorsunuz. Nesnenin metotlarından biri olan adi_belirle fonksiyonuna bir değer veriyoruz: "Zinde"; nesne oluşturulurken yazılmış olan bu fonksiyon ise aldığı değeri, kendi ait olduğu nesnenin bir değişkenine kaydediyor:

```
function adi_belirle ($n) {
$this->adi = $n;
}
```

Bu ve diğer fonksiyonlarda kullandığımız **"$this->" ifadesi,** kendisine ulaştırılan değeri bir parametre olarak kullanıyor ve **"$adi"** değişkenine yazıyor. **"this" (bu)** kelimesi nesnenin o anda oluşturulmakta olan örneğine göndermede bulunur. "->" işlemcisini kullanarak istediğimiz nesnenin istediğimiz metoduna veya parametresine değer gönderebiliriz. Bir nesnenin yeni bir örneğini oluşturduğumuz zaman bu örneğin bütün parametrelerini sağlamak veya bütün metotlarını kullanmak zorunda değiliz.

Yukarıdaki örneğe göre yeni örnekler oluşturarak en az dört öğrencinin not ortalamasını hesap edebilir misiniz?

2.1.4. PHP'de Dosya Oluşturma

Artık bir HTML formuna girilen verilere nasıl erişeceğinizi ve bunları nasıl düzenleyeceğinizi bildiğinizi düşünüyoruz. Bu bilgileri sonra kullanmak, değerlendirmek, üzerlerine yenilerini eklemek, vb. işlemler için PHP'de dosyalamanın nasıl yapıldığına bakacağız. Veri depolamanın temelde iki yolu vardır: Düz dosyalarda (flat file) veya bir veri tabanı içinde. Düz dosyanın birçok biçimi bulunabilir. Anca genelde düz dosya dediğimizde basit bir text (metin) dosyasını kastederiz.

PHP ile bir dosyaya bağlanıp o dosya içerisindeki verileri okuma-yazma işlemlerinde PHP'ye dosyanın yerini bildirmek gerekir. Bir örnek verelim:

```
<?php
$dosya="dosya.txt";
        If ( file_exists($dosya) ) {
        echo "Dosya var";
        }
        Else {
        Echo "dosya yok";
}
?>
```

Bu komut satırlarını çalıştırdığınızda ekranda "**dosya yok"** mesajı okunacaktır. Çünkü belirtilen dizinde dosya.txt adında bir dosya yoktur. Dizin içerisinde dosya olup olmadığını test etmek için file_exists komutu kullanılır. Bazı durumlarda dosya olmadığında bu komut hata numaraları verir. Dolayısıyla hata numaralarının ekranda gözükmemesi için bu komutun başına @ koymakta yarar vardır.

Bazı Dosya Dizin Komutları			
is_file	Dosya mı?	copy	Dosya kopyalama
is_dir	Dizin mi?	diskfreespace	Disk üzerindeki boş alan
is_readable	Okunabilir mi?	disk_total_space	Disk üzerindeki dolu alan
is_writeable	Yazılabilir mi?	touch	Dosya oluşturma
filesize	Dosyanın boyutu	unlink	Dosya silme

Tablo 2.1: Dosya dizin komutları

PHP`de bir dosyanın içeriğini alarak sayfalarımızda kullanma veya bir dosyanın içeriğini değiştirmek gibi işlemler için önce dosyanın açılmış olması gerekir. Bunu gerçekleştiren **fopen()** fonksiyonudur. Dosyayı açtığınızda, onu nasıl kullanmak istediğinizi belirtmeniz gerekir. Sunucu üzerindeki işletim sistemi dosyayla ne yapmak istediğinizi bilmek isteyecektir. Bir dosyayı açmak için üç seçim yapmanız gerekir:

- Bir dosyayı sadece okumak için, sadece yazmak için veya hem okumak hem de yazmak için isteyebilirsiniz.

- Eğer bir dosyaya yazıyorsanız dosyanın mevcut içeriğinin üzerine yazabilirsiniz veya dosyanın sonuna yeni veri ekleyebilirsiniz. Ayrıca dosya zaten mevcutsa bunun üzerine yazmak yerine durumu kabullenip programınızı sonlandırmayı da tercih edebilirsiniz.
- Eğer binary (ikilik) dosyalar ve metin dosyaları arasında ayrım yapan bir sistem üzerindeki bir dosyaya yazmaya çalışıyorsanız bunu belirtmek isteyebilirsiniz.

Fopen() fonksiyonu, bu üç seçeneğin kombinasyonlarını destekler. Bunun için aşağıdaki tabloda fopen() için dosya kipleri verilmiştir.

Kip	Kipin Adı	Anlamı
r	Okuma(read)	Dosyayı, baştan itibaren okuma için açar.
r+	Okuma	Dosyayı, baştan itibaren okuma ve yazma için aç.
w	Yazma(write)	Dosyayı, başından itibaren yazma için aç. Eğer dosya mevcutsa mevcut içeriği sil. Eğer mevcut değilse oluşturmaya çalış.
w+	Yazma	Dosyayı, başından itibaren yazma ve okuma için aç. Eğer dosya mevcutsa mevcut içeriği sil. Eğer mevcut değilse oluşturmaya çalış.
x	Tedbirli yazma	Dosyayı, başından itibaren yazma için aç. Eğer dosya mevcutsa, açılmayacak, **fopen() false** sonucunu döndürecek ve PHP bir uyarı mesajı gönderecektir.
x+	Tedbirli yazma	Dosyayı, başından itibaren yazma ve okuma için aç. Eğer dosya mevcutsa açılmayacak, **fopen() false** sonucunu döndürecek ve PHP bir uyarı mesajı gönderecektir.
a	Ekleme (Append)	Dosyayı, eğer mevcutsa içeriğin sonundan itibaren sadece ekleme yapmak (yazmak) için aç. Eğer mevcut değilse oluşturmaya çalış.
a+	Ekleme	Dosyayı, eğer mevcutsa içeriğin sonundan itibaren sadece ekleme yapmak (yazmak) ve okumak için aç. Eğer mevcut değilse oluşturmaya çalış.
b	İkilik (Binary)	Diğer kiplerden biriyle birlikte kullanılır. Eğer dosya sisteminiz ikilik dosyalar ve metin dosyaları arasında ayrım yapıyorsa bu kipi kullanmak isteyebilirsiniz.
t	Metin (Text)	Diğer kiplerden biriyle birlikte kullanılır. Bu kip yalnızca Windows sistemlerinde mevcut olan bir seçenektir.

Tablo 2.2. fopen() için dosya kipleri

Fopen() fonksiyonuyla bir dosya okumak **(r)**, yazdırmak **(w)** veya ek yapmak **(a)** için açabiliriz. Bu fonksiyon, dosyanın başarıyla açılması hâlinde bir tam sayı verecektir. PHP programlarımızda, açılan dosyanın mutlaka ona işaret eden bir değişkene (file pointer) bağlı olması gerekir. Daha sonra bu dosya ile ilgili bütün işlemleri bu işaret değişkeni ile yaparız.

Örnek:

```
$dosya = fopen( "bir_dosya. txt" , " r ");
```

PHP, bu dosyayı sadece okumak amacıyla açacak ve fonksiyondan dönen değeri $dosya değişkenine atayacaktır. Olmayan bir dosyayı açmak istediğimiz zaman PHP hata mesajı verir. Bir dosyayı yazmak amacıyla açacağımız zaman bu kodu şöyle yazarız:

```
$dosya = fopen( "bir_dosya.txt" , "w" );
```

Olmayan bir dosyayı yazmak amacıyla açmak istediğimizde PHP önce bu dosyayı oluşturur. Bir dosyaya ek yapmak istediğimiz zaman ise kodumuz şu şekilde yazılır:

```
$dosya = fopen( "bir_dosya.txt" , " a" );
```

Olmayan bir dosyayı ek yapmak amacıyla açmak istediğimizde PHP hata mesajı verir.

PHP'de genelde dosya işlemlerinde okunacak dosyanın metin bilgisi içerdiği varsayılır. Dosyaya bir tanımlama kurulduktan sonra dosyadan satır satır bilgi okuma işlemi gerçekleştirilebilir. Satır satır bilgi okuma işleminde dosya sonuna gelindiğinin anlaşılabilmesi için feof() komutu kullanılır. Her dosyanın sonda End Of File (eof) adında özel bir işaret vardır. Eğer okunulan satırda bu işaret varsa dosya okuması bitmiş demektir. Feof() komutu açılan dosya tanımlamasında okunulan satır eğer eof işaretini içeriyorsa doğru, aksi durumda yanlış sonucu döndürür. Açılan dosyadan satır okumak için fgets() komutu kullanılır. Bu komutta, bir satırda okunacak maksimum karakter sayısı da bildirilmelidir.

```
$dosya=fopen("okunacakdosya.txt", " r" );
while (!feof($dosya))
        {
        $satir=fgets($dosya,4096);
        echo $satir;
        }
fclose($dosya);
```

Yukarıdaki kodda okunacakdosya.txt adındaki dosya açılarak while döngüsüyle eof karakteri okunana dek fget() komutuyla satır çekilir. Okunan bu satır $satir adındaki string değişkene aktarılır ve echo komutuyla ekrana yazdırılır. En sonda yer alan fclose() fonksiyonu ile daha önce tanımlanmış olan $dosya tanımlayıcısı kapatılarak dosya kapatılır.

Dosyanın satır satır okunarak ekrana basılması yerine tek bir komutla dosyayı okuyarak ekrana yazmak da mümkündür. Bu işlem için readfile() fonksiyonu kullanılır. Bu durumda dosyaya bir tanımlayıcı kurmaya gerek kalmaz.

```
readfile("okunacak.txt");
```

2.1. Nesne Düzenleme

2.2.1. PHP'de Dosya Okuma

PHP'de bir dosyanın içindeki bilgileri çekmek için üç farklı fonksiyon vardır: **fgets(), fread(), fgetch()**

fgets()

Bu fonksiyon parametre olarak daha önce açılmış olan dosyaya ait değişkeni ve asgari okuyacağı byte miktarını alır. Fonksiyon, verdiğimiz uzunluk ölçüsüne ulaşmadan önce dosyada yeni satır işareti görürse ya da dosya sonuna ulaşırsa okumaya son verir. Genelde dosyadan satır satır okuma yaptığımız durumlarda dosyanın tamamını okumak için bir döngü kullanırız. Döngü koşulu olarak dosya sonuna ulaşılıp ulaşılmadığını **feof()** fonksiyonu ile kontrol etmemiz gerekir.

Root klasörümüzün (www klasörü) içinde php dizininde bulunan ornek.txt dosyası olduğunu düşünelim. Eğer bu dosya yoksa oluşturabilirsiniz.

fgets()

```
<?
$dizin="php";
$dosya_adi="ornek.txt";
if ($dosya=fopen($dizin."/".$dosya_adi, " r "))
{    echo  "Dosya Adı:".$dosya_adi."<br>";
   $i=0;
   while(!feof($dosya))   //dosya sonu gelmediği müddetçe
   {   $i++;
      $satir=fgets($dosya,1024);
      print $i.".Satır: ".$satir."<br>";
   }
}
else
   print "Dosya açılamadı: ".$dosya_adi;
?>
```

fread()

Dosyadan satır satır değil de blok okumak istediğimiz durumlarda bu fonksiyonu kullanırız. fgets() ile aynı parametreleri alır, farklı olarak yeni satır karakteri görünce durmaz. Bu sayede dosyadan tamamını ya da belirttiğimiz uzunlukta bir bloğunu okuyabiliriz.

```
<?
$dizin="php";
$dosya_adi="ornek.txt";
```

```
if ($dosya=fopen($dizin."/".$dosya_adi, "r"))
{
   print "Dosya Adı: ".."<br>";
   print "Dosya Boyu: ".filesize()." Byte<br>";

   $blok=fread($dosya,filesize($dosya_adi));
   print "Dosya İçeriği: <br>";
   print nl2br($blok);
}
else
   print "Dosya açılamadı: ".$dosya_adi;
?>
```

Bu örnekteki **nl2br()** fonksiyonu metin işlemleri konusunda da göreceğimiz bir fonksiyondur. Parametre olarak aldığı metindeki **yeni satır** karakterlerini HTML etiketi olan ve tarayıcı ekranında yeni bir satıra geçilmesini sağlayan <**br**> ifadesine dönüştürmeye yarar.

fgetc()

Bu fonksiyon dosyadan her defasında bir karakter okutmak için kullanılır. Daima 1 byte veri okuyacağı için parametre olarak sadece dosya değişkenini alır.

```
<?
$dizin="php";
$dosya_adi="ornek.txt";
if ($dosya=fopen($dizin."/".$dosya_adi, "r"))
{
   print "Dosya Adı: ".$dosya_adi."<br>";
   $i=0;
   while(!feof($dosya))   //dosya sonu gelmediği müddetçe
   {
      $i++;
      $krk=fgetc($dosya);
      print $i.".Karakter: ".$krk."<br>";
   }
}
else
   print "Dosya açılamadı: ".$dosya_adi;
?>
```

2.2.2. PHP'de Dosyaya Yazma

PHP'de sabit diskteki dosyaya bilgi yazma işlemi, okuma işlemine oldukça benzer. Önce dosya açılarak bir tanımlayıcı oluşturulur. Daha sonra satır satır bilgiler yazılır veya yazma işlemi bitince de dosya kapatılır. Bir dosyaya yazma ya da ek yapma PHP için aynı şeydir; sadece dosyaların açılışında fark vardır. Dosya, **w** (write) parametresi ile açılmışsa

yazma işlemi dosyanın en başından başlar, **a** (append) parametresi ile başlamışsa yazma işlemi dosyanın sonundan başlar. Bir dosyayı yazmak amacıyla açmak için:

```
$dosya = fopen( "bir_dosya.txt" , "w" ) or die ("Dosya açılamıyor!") ;
```

Dosyaya yazma fonksiyonları **fwrite()** ve **fputs()**'dur. Bu iki fonksiyon tamamen aynı işi yapar. Aralarında bir fark yoktur. Genel kullanım şekilleri:

```
fputs($dosya,"Dosyaya yazılacak yada eklenecek ifade..");
```

Örnek 1:

```
<?
$dosya=fopen("yazilacakdosya.txt","w");
$i=0;
while ($i<10)
  {
  $bilgi=$i.".kayıt\n";
  fwrite($dosya,$bilgi);
  $i++;
}
fclose($dosya);
?>
```

PHP, bu kodla yazilacakdosya.txt adında bir dosyaya tanımlayıcı kurar. Dosyanın tanımlayıcı tür bilgisi olarak **"w"** kullanıldığından dosya tanımlayıcısı çağrıldığında yoksa oluşturulur, eğer varsa dosya silinerek yenisi oluşturulur. Daha sonra while döngüsüyle $i değişkeninin 0'dan 10'a kadar aldığı değerler fwrite() komutuyla dosyaya yazılır. İşlem bitince fclose() komutu çağrılarak dosya kapatılır.

Örnek 2:

Sitemize gelen ziyaretçilerin IP'lerini ve ziyaret saatlerini bir dosyada tutacak bir fonksiyon yazalım. IP'leri kaydedeceğimiz dosyanın giderek şişmemesi için belli bir boyutu aştığında dosyayı tekrar sıfırdan oluşturmaya başlayalım.

```
<?
function LogTut()
{
   global $REMOTE_ADDR;   // Siteye bağlanan makinenin IP adresini tutan
sunucu değişkeni

   $logDosyasi="/tmp/site_log.txt";
   $logSiniri=1024*1024; // 1MB

   // Tarih ve saat bilgisini yazdırmak için date() fonksiyonunu kullanıyoruz
```

```
    $tarihSaat=date("d-m-Y H:i:s"); // Örn. 21-06-2002 14:30:05

    /* Log dosyası, dosya yoksa hata oluşmaması için ya da varsa ve boyutu
1 MB'yi aşmıssa sıfırlanması
    için "w" modunda, diğer durumlarda "a" modunda açılacaktır */
    $mod=(!file_exists($logDosyasi) or (file_exists($logDosyasi) and sizeof($l
ogDosyasi)>$logSiniri)) ? "w" : "a" ;

    $dosya=fopen($logDosyasi,$mod);

    /*Dosyamızdaki satırların düzenli olması için saat ile IP arasına TAB
karakteri,
    IP'den sonra da satır başı karakeri ekliyoruz...*/
    fputs ($dosya,$tarihSaat."t".$REMOTE_ADDR."n");

    fclose($dosya);
}
?>
```

Örnek 3:

```
 <?php
$dosya_adi = "bir_dosya.txt";
$dosya = fopen ($dosya_adi , "w") or die ("Dosya açılamadı!");
$metin = "Bu satır dosyaya yazılacak: Merhaba Dünya!n";
fwrite ( $dosya , $metin ) ;
fputs ( $dosya , "Bu satır ise sonradan eklenecek" ) ;
fclose ($dosya);
?>
```

Bu programı çalıştırdığınızda **bir_dosya. txt** adlı dosyada mevcut bütün içerik silinecek ve yerini **$metin** değişkeninin içerdiği "Bu satır dosyaya yazılacak: Merhaba Dünya!" yazısı ile "Bu satır ise sonradan eklenecek" cümlesi alacaktır. Her iki metnin sonunda da yeni satır işareti bulunduğuna dikkat ediniz. **Bu programda dosya açma komutundaki "w" parametresini siler, yerine "a" yazarsanız** bu metinlerin dosyanın içeriğine eklendiğini görebilirsiniz.

2.2.2. Dosya Kapatmak

Dosyalar işlendikten sonra (okuma, yazma, ekleme vb.) onu kapatmak gerekir. Bunu, fclose() fonksiyonunu kullanarak aşağıdaki gibi yapabilirsiniz:

```
flose($dosyaadi);
```

Bu fonksiyon eğer dosya başarıyla kapanmışsa true, kapanmamışsa false sonucunu döndürür.

2.2.3. Dosya Silmek

PHP ile yapabileceğimiz önemli dosya işlemlerinin başında olmayan bir dosyayı oluşturmak ve olan bir dosyayı silmek gelir. PHP`nin dosya oluşturma komutu touch() fonksiyonudur. Bu fonksiyona oluşturulmasını istediğimiz dosyanın adını vermemiz gerekir.

Örnek:

```
<?php
$dosya_dizin = "php";
touch ($dosya_dizin."/yeni_belge.txt");
print ("yeni_belge adlı bir dosya oluşturuldu!");
?>
```

Bu programı kişisel web sunucuda denerken yeni dosyanın oluşturulacağı dizin olarak "/" işaretiyle sadece kök dizini belirtirseniz dosya C: diskinde kök dizinde oluşturulur. Bu programı gerçek sunucuda çalıştırabilmek için yazma/okuma izni bulunan ve web sunucunun erişebileceği bir dizinin adını vermeniz gerekir.

Örnek:

```
<?php
$dosya_dizin = "/wwwroot/mycgiserver.com/members/uNhM13Qnm/";
touch ("$dosya_dizin. "/yeni_belge.txt");
print ("yeni_belge adlı bir dosya oluşturuldu!");
?>
```

Bu komutla oluşturacağınız dosya içi boş bir metin dosyası olacaktır. Eğer belirttiğiniz dizinde bu adı taşıyan bir dosya varsa PHP dosyanın içeriğine dokunmayacak, fakat dosyanın erişim ve değişim tarihlerini değiştirecektir.

PHP ile mevcut bir dosyayı silmek için **unlink()** fonksiyonunu kullanırız. Bu fonksiyon da silinecek dosyanın adı ile birlikte yolunu ister.

Örnek:

```
<?php
$dosya_dizin = "php";
unlink ($dosya_dizin."/yeni_belge.txt");
print ("yeni_belge adlı dosya silindi!");
?>
```

2.2.4. Dosyaya Bilgi Eklemek

Bir dosyaya yazma veya ek yapma, PHP açısından aynı işlemdir; sadece dosyaların açılışında fark vardır. Bir dosyayı yazmak amacıyla açmak için:

```
$dosya = fopen( "bir_dosya.txt" , "w" ) or die ("Dosya açılamıyor!") ;
```

Bilgi eklemek amacıyla açmak için ise

```
$dosya = fopen( "bir_dosya.txt" , "a" ) or die ("Dosya açılamıyor!") ;
```

kodunu yazmamız gerekir. Daha sonra yapılacak yazma ve ekleme işlemlerinin farkı, `w` parametresi ile açılan dosyaya yazma işlemi en başından başlar ve devam eder; "**a**" parametresi ile açılan dosyaya yazma işlemi ise en sondan başlar ve devam eder.

PHP`nin bir dosyaya metin yazdırma fonksiyonları olan **fwrite( ) ve fputs( )** aynı biçimde yazılır ve aynı işlevi yerine getirir; aralarında kesinlikle fark yoktur.

Örnek:

```
<?php
$dosya_adi = "bir_dosya.txt";
$dosya = fopen ($dosya_adi , "w") or die ("Dosya açılamadı!");
$metin = "Bu satır dosyaya yazılacak: Merhaba Dünya!n";
fwrite ( $dosya , $metin ) ;
fputs ( $dosya , "Bu satır ise sonradan eklenecek" ) ;
fclose ($dosya);
?>
```

Bu programı çalıştırdığınızda, **bir_dosya.txt** adlı dosyada mevcut bütün içerik silinecek ve yerini **$metin** değişkeninin içerdiği "Bu satır dosyaya yazılacak: Merhaba Dünya!" yazısı ile "Bu satır ise sonradan eklenecek" cümlesi alacaktır. Her iki metnin sonunda da yeni satır işareti bulunduğuna dikkat ediniz. Bu metinlerin dosyanın içeriğine "a" parametresi ile eklendiğini görebilirsiniz.

2.2.5. Dosyayı Kilitlemek

Web sunucusundaki dosyalarımızla sadece bir kişi işlem yapıyor olsa idi bir sorun olmazdı. Ne var ki bir web sitesine aynı anda birden fazla kişi erişebilir ve dosyalarla işlem yapan programları çalıştırıyor olabilir. Bu, PHP`nin dosya işlemlerine engel olabilir.

Bu sebeple işlem için açacağımız bir dosyayı önce kilitlemek, yerinde bir önlem sayılır. Bunu, flock() fonksiyonu ile yaparız; bu fonksiyona kilitlemek istediğimiz dosyanın işaret değişkeninin adını ve kilit türünü belirten endeks sayısını parametre olarak yazarız.

İki müşterinin bir ürünü aynı anda satın almaya çalıştıkları bir durum düşününüz (Trafiği yoğun bir web sitesinde sık rastlanan bir durum olabilir.). Müşterilerden biri fopen() fonksiyonunu çağırdı ve yazmaya başladı. Diğeri de fopen() ‘ı çağırıp yazmaya başladığında ne olacak? Dosyanın son içeriği nasıl olacak? İlk siparişi mi, ikinci siparişi mi takip edecek,

yoksa tam tersi mi olacak? Siparişlerden sadece biri mi alınacak? Yoksa iki sipariş birbirine mi karışacak? Bunun cevabı, işletim sisteminize bağlı olsa da bunu bilmek çoğunlukla imkânsızdır.

Bu gibi sorunlardan kaçınmak için dosya kilitleme işlemini kullanabilirsiniz. Bu PHP'de flock() fonksiyonu kullanılarak gerçekleştirilir. Bu fonksiyon bir dosya açıldıktan sonra, ancak dosyadan veri okunmadan ya da dosyaya veri yazılmadan önce çağırılmalıdır. Flock() kullanımı:

```
bool flock (resource  fp, int operation [ int $wouldblock] )
```

Burada, açık bir dosyadaki bir işaretçiyi ve ihtiyaç duyduğunuz kilit tipini temsil eden bir sabit değeri kullanmanız gerekir. Kilit başarıyla uygulandıysa true, aksi hâlde false sonucunu verir. İsteğe bağlı olan üçüncü parametre, kilidin uygulanması geçerli işlemin bloke edilmesine (beklemek zorunda kalmasına) neden olduğu takdir de true değerini içerir.

İşlemin olası değerleri aşağıdaki tabloda gösterilmiştir:

İşlem Değeri	**Anlamı**
LOCK_SH (daha önce 1)	Okuma kilidi. Dosya başka okuyucularla paylaşılabilir.
LOCK_EX (daha önce 2)	Yazma kilidi. Bu işlem özeldir. Dosya paylaşılamaz.
LOCK_UN (daha önce 3)	Mevcut kilit kaldırılır.
LOCK_NB (daha önce 4)	Bu kilit uygulamaya çalışırken bloke edilmeyi engeller.

Tablo 2.3: flock ( ) işlem değerleri

Eğer flock () kullanacaksanız, bu fonksiyonu dosyayı kullanan tüm script'lere eklemeniz gerekir. Aksi hâlde işe yaramayacaktır.

Örnek:

```
$fp=fopen("$DOCUMNET_ROOT/../orders.txt",'ab');
flock($fp,LOCK_EX); // dosyaya kilidi ekle
fwrite ($fp, $outputstring);
flock ($fp, LOCK_UN); // yazma kilidini kaldır
fclose ($fp);
```

Örnek:

```
<?php
$dosya_adi = "bir_dosya.txt";
$dosya = fopen ($dosya_adi , "w") or die ("Dosya açılamadı!");
flock ( $dosya , 2); // dosyayı kilitle
$metin = "Bu satır dosyaya yazılacak: Merhaba Dünya!n";
fwrite ( $dosya , $metin ) ;
fputs ( $dosya , "Bu satır ise sonradan eklenecek" ) ;
```

```
flock ( $dosya , 3); //dosyayı kilidini aç
fclose ($dosya);
?>
```

Bu fonksiyon ile kullanabileceğimiz endeks parametreleri şunlardır:

- Paylaşım diğer işlemlerin dosyayı paylaşmalarına imkân verir.
- Tüm diğer işlemlerin dosya ile işlem yapmasına engel olur.
- Serbest dosyanın 1 veya 2 olan kilidini kaldırır.

Bir dosya, herhangi bir PHP programı tarafından kilitlendiği anda, aynı dosyayı daha sonra kilitlemeye kalkan diğer programlar kendilerinden önce konulmuş kilide saygı gösterir.

2.2.6. Nesnelerle Dosya Dizin İşlemleri

PHP 5 ile gelen yeni nesne yönelimini kullanarak aşağıdaki saf PHP kodlarını kullanarak bir nesne oluşturalım. Bu nesnede temel dosya ve dizin işlemlerini yürüten komut yapısı bulunmaktadır. Tüm dosya ve dizin işlemlerini buradaki örneği kullanarak geçekleştirebilirsiniz.

```
<?php
class dosya extends Exception
{
/*
Sabit degişkenleri bu alanda tanımlıyoruz
*/
public $dizin;
public  $dosya;
public  $dizi = array ( );
/*
Server ana dizini nesneye tanımlatıp tüm nesne içerisinde kullanıyoruz.
*/
public  function _construct ($dizin)
{
   $this->dizin = $dizin;
}
/*
dosya olup olmadığını kontrol eder
*/
public function dosya Varmi( )
{
try
{
  İf ( ! @ file _ exists ($this ->dizin. $this->dosya)  )
  throw new Exception ( false);
}
catch ( Exception $error)
```

```
{
Return (false) ;
  }
}
/*
dosya aç
*/
pubic function dosyaAc ($parametre)
(
  Try
  {
  Return fopen ($this->dizin. $this->dosya ) ;
  }
  Catch ( Exception $error )
  {
    die ("{$error->getMessage( )}");
   }
/*
Dosyaya yaptığımız bağlantıyı koparma
*/
public function dosya Kapa( )
{
return @fclose (dosya: : dosyaAc( ) );
}
/*
dosya boyutu
*/
public function dosyaBoyutu ( )
{
return file size( $this->dizin.$this->dosya);
}
/*
Dosya oku
*/
Public function dosyaOku($parametre)
}
try
  {
      $ac = dosya : : dosyaAc ($parametre);
       İf   (  !   $ac  )
              thorow new Exception(false);
 }
else
{
               while (! Feof($ac) )
              {
              echo fread ($ac,  dosya : : dosyaBoyutu ( ) ) ;
               }
```

```
     }
   dosya :  : dosyaKapa ( ) ;
}
catch (Exception  $error  )
{
     die ($error);
  }
}
/*
dosyaYaz
*/
Public function dosyaYaz ($sonuc,$parametre)
{
try
  {
   $ac= dosya : : dosyaAc ($parametre);
   İf      (  ! fwrite ($ac , $sonuc)  )
           Throw new Exception (false);
           @chmod ($ac, 0777);
        return true;
   }
   catch ( Exception $error  )
   {
    die ("{$error->getMessage ( )}");
    }
    dosya : : dosyaKapa ( );
 }
/*
dizin var mı? Yoksa dizin oluştur
*/
  try
  {
    İf  (  ! @is _dir ($this->dizin)  )
    {
    İf  ( ! mkdir ($this ->dizin,0777 )   )
         throw new Exception ( true ) ;
       else
         chmod ($this->dizin, 0777 ) ;
     }
}
catch ( Exception $error )
  {
return ("{$error->getMessage ( ) }");
  }
}
#dizin varmı?
public function dizinOluştur2 ($dizin)
  {
```

```
  try
   {
    İf  (  ! @is|_dir($dizin)   )
   {
    İf  (  ! @mkdir($dizin,0777 )  )
  throw new Exception (true );
    else
              chmod($dizin,0777);
      }
}
catch ( Exception $error )
 {
return ("{$error->getMessage ( ) }");
  }
}
/*
dizin varmı?
*/
dizin varmı?
*/
Public function dizin Varmi ( )
{
   try
  {
   if (   is_dir ($this->)  )
          throw new Exception (false)
         $sonuc = true;
         }
         catch  ( Exception $error   )
         {
            die ("{$error->getMessage ( ) }");
         }
         return $sonuc;
}
/*
Dosya kopyalama
*/
public function dosyaTasima( $dosya1, $dosya2 )
{
  $d1 =$this->dizin.$dosya1;
  $d1 =$this->dizin.$dosya2;
if  ( ! is_dir($this->dizin)  )
{
try
}
       if ( exec( "mv  $d1 $d2", $a, $b ) )
       throw new Exception (false);
       return true;
```

```
}
catch ( Exception $error )
{
      return ( "{$error->getMessage  ( ) }");
 }
/*

dosya silme. Biz burada direk UNIX komutu çalıştırarak
bu işi yapıyoruz, isteyenler unlink komutunu kullanabilirler
*/
public function dosyaSil ($dosya)
{
  try
 {
      if  ( exec ("rm $dosya")  )
      throw new Exception (false);
       return true;
}
catch ( Exception $error )
{
  return ("{$error->getMessage ( ) }");
  }
}
}
?>
```

Bir iki örnek vererek yukarıdaki nesneyi nasıl kullanacağınıza bakalım;

Dosya var mı?

```
<? Php
$dosya = new dosya( $_SERVER[`DOCUMENT_ROOT'] );
$dosya -> dosya = "dosya.php";
If ( $dosya->dosyavarmi( ) )
  Echo "dosya var. ";
Else
 Echo "dosya yok";
?>
```

Dosya Yazma

```
<?php
$dosya = new dosya( $_SERVER[`DOCUMENT_ROOT'] );
$dosya -> dosya = "dosya.php";
$metin="metinsel içerik";
$dosya->dosyayaz( $metin, "w");
?>
```

Dosya Silme

```
<? Php
$dosya = new dosya( $_SERVER[`DOCUMENT_ROOT’] );
$dosya -> dosyasil("dosya.php");
?>
```

Dizin Yoksa Oluştur

```
<? Php
$dosya = new dosya( $_SERVER[`DOCUMENT_ROOT’] );
$dosya -> dizin = "veri";
$dosya -> dizinolustur( );
?>
```

UYGULAMA FAALİYETİ

Ders öğretmeniniz Internet Programcılığı dersi sınav notlarını internetten yayımlamak istiyor. Bunun için normal metin dosyasında öğrenci numaraları ve notları olacak, öğrenci numarasını girdiğinde browser'da öğrenci sınavdan almış olduğu notu görecek.

İşlem Basamakları	Öneriler
➢ Öğrenci numara ve notlarını Excel'de hazırlayınız. ➢ Bu Excel dosyasını "csv" uzantılı olarak (virgülle ayrılmış veri dosyası) kaydediniz. ➢ Dosya okuma modunda açılması gerekli komutları oluşturunuz. ➢ Dosyaya en sonunda kapatmak için gerekli komutları oluşturunuz. ➢ Birden çok öğrencinin aynı anda dosyaya ulaşmak isteği durumda sorun çıkmaması için gerekli dosya kilitleme komutlarını yazınız.	➢ Arama işlemlerinde if ( ), feof ( ) komutu , while döngüsü gibi PHP yapılarını kullana bilirsiniz. ➢ Arama işlemi için form yapısını ve input komutlarını kullanacaksınız.

ÖLÇME VE DEĞERLENDİRME

Bu bölümde ikinci öğrenme faaliyetinde öğrendiğiniz bilgileri ölçebileceksiniz. Ölçme sorularını dikkatlice okuyarak cevaplandırınız.

OBJEKTİF TEST (ÖLÇME SORULARI)

Aşağıda çoktan seçmeli bulunmaktadır. Soruları okuyarak doğru bulduğunuz seçeneği işaretleyiniz. Test soruları bitirince Cevap Anahtarı ile karşılaştırınız.

1. Aşağıdakilerden hangisi PHP'de sınıf oluşturma komutudur?
A) object B) class C) sinif D) new

2. PHP'de nesne hangi komutla oluşturulur?
A) object B) class C) new D) this

3. PHP'de dosya oluşturmak için aşağıdaki fonksiyonlardan hangisi kullanılır?
A) fopen ( ) B) fcreat ( )
C) fgetch ( ) D) fnew ( )

4. PHP'de herhangi bir amaçla açılan dosyaların kapatılması gerekir. Bunun için hangi fonksiyon kullanılır?
A) fopen ( ) B) fclose ( ) C) fgets ( ) D) readfile ( )

5. PHP'de bir dosyanın içindeki bilgileri çekmek için kullanılan fonksiyon hangisidir?
A) fgets ( ) B) fread ( ) C) fgetch ( ) D) Hepsi

6. PHP'de işlem yapılan dosyanın sonuna gelinip gelinmediğini kontrol eden fonksiyon aşağıdakilerden hangisidir?
A) eof ( ) B) feof ( ) C) Fgets ( ) D) fopen ( )

7. PHP'de dosya yazılma işlemi için hangi parametre ile açılmalıdır?
A) r B) b C) x D) w

8. PHP'de bir dosya silinecekse aşağıdaki komutlardan hangisi kullanılır?
A) del ( ) B) fclose ( )
C) unlink ( ) D) kill ( )

9. Web ortamında bir dosyala birden çok kullanıcı yazma vb. yapması gerektiğinde karışıklığa meydan vermemek için dosyalara kilitleme işlemi yapılır. Kilitleme işlemi yapan fonksiyon aşağıdakilerden hangisidir?
A) fclose ( ) B) flocked ( )
C) lock ( ) D) flock ( )

10. Aşağıdaki seçeneklerden hangisinde bir dosya yazılmak için açılmıştır?
A) fopen("bir_dosya.txt" , "w")
B) fopen("bir_dosya. txt" , "r”);
C) fopen("bir_dosya. txt" , “b”);
D) fopen("bir_dosya. txt" , “t”);

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Sınav içinde cevaplandıramadığınız, yanlış cevaplandırdığınız veya kendinizi bilgi bakımından eksik hissettiğiniz sorular için ilgili konulara tekrar dönünüz.

MODÜL DEĞERLENDİRME

OBJEKTİS TEST (ÖLÇME SORULARI)

Bu kısımda öğrenme faaliyetlerinde öğrendiğiniz bilgileri kullanarak uygulama faaliyetlerinde gösterdiğiniz becerilerin tümünü kapsayan bir ölçme işlemi yapılacaktır. Bu performans değerlendirme işlemini öğretmeninizin kontrolünde gerçekleştiriniz. Doğru olarak düşündüğünüz seçeneği işaretleyiniz.

Aşağıda çoktan seçmeli sorular bulunmaktadır. Soruları okuyarak doğru bulduğunuz seçeneği işaretleyiniz.

1. Web ortamında kullanıcıdan bilgi almak için kullanılan formlarda hangi metotlar kullanılır?
A) Put B) Post C) OutPut D) Send

2. Form bilgileri hangi tag komutları kullanılır?
A) <form> … </form> B) <form> … </script>
C) <script> … </form> D) Hiçbiri

3. Form metotlarından hangisi daha az güvenilirdir?
A) Text B)Submit C) Get D) Post

4. Aşağıdakilerden hangisi input type'ındandır?
A) input B) Submit C) Post D) eregi

5. Formlarda düz metin girişi için hangi inptu type'ı kullanılır?
A) Check Box B) Password C) Select Box D) Text

6. Formlarda tek seçenekli durumlarda kullanılan input type'ı hangisidir?
A) Option B) Radio C) Select Box D) Check Box

7. Formları hazırladıktan sonra formlara girilen bilgileri göndermekten vazgeçmek için hangi inptu type'ı kullanılır?
A) Submit B) Value C) Check Box D) Reset

8. Bir değişken içinde değer bulunup bulunmadığını sınayan PHP fonksiyonu aşağıdakilerden hangisidir?
A) empty ( ) B) strlen ( ) C) isset ( ) D) eregi ( )

9. Formlarda girilecek değerin karakter sayısını sınırlamak için hangi PHP fonksiyonu kullanılır?
A) empty ( ) B) strlen ( ) C) isset ( ) D) eregi ( )

10. Değişkenim içerisi boş olduğunda (değer girilmemişse) TRUE değeri gönderen PHP fonksiyonu aşağıdakilerden hangisidir?
A) eregi ( ) B) strlen ( ) C) isset ( ) D) empty ( )

11. Aşağıdakilerden hangisi PHP'de dizinde dosya olup olmadığını kontrol eden komuttur?
A) object B) class C) file_exists D) new

12. PHP'de nesne hangi komutla oluşturulur?
A) object B) class C) new D) this

13. PHP'de dosya oluşturmak için aşağıdaki fonksiyonlardan hangisi kullanılır?
A) fopen ( ) B) fcreat ( ) C) fgetch ( ) D) fnew ( )

14. PHP'de herhangi bir amaçla açılan dosyaların kapatılması gerekir. Bunun için hangi fonksiyon kullanılır?
A) fopen ( ) B) readfile ( ) C) fgets ( ) D) fclose ( )

15. PHP'de bir dosyanın içindeki bilgileri çekmek için kullanılan fonksiyon hangisidir?
A) fgets ( ) B) fread ( ) C) fgetch ( ) D) Hepsi

16. PHP'de işlem yapılan dosyanın sonuna gelinip gelinmediğini kontrol eden fonksiyon aşağıdakilerden hangisidir?
A) feof ( ) B) eof ( ) C) Fgets ( ) D) fopen ( )

17. PHP'de dosya yazılma işlemi için hangi parametre ile açılmalıdır?
A) r B) b C) x D) w

18. PHP'de bir dosya silinecekse aşağıdaki komutlardan hangisi kullanılır?
A) del ( ) B) fclose ( ) C) touch ( ) D) unlink ( )

19. Web ortamında bir dosyala birden çok kullanıcı yazma vb. yapması gerektiğinde karışıklığa meydan vermemek için dosyalara kilitleme işlemi yapılır. Kilitleme işlemi yapan fonksiyon aşağıdakilerden hangisidir?
A) fclose ( ) B) flocked ( ) C) lock ( ) D) flock ( )

20. Aşağıdaki seçeneklerden hangisinde bir dosya, yazılmak için açılmıştır?
A) fopen("bir_dosya. txt" , "b");
B) fopen("bir_dosya. txt" , "r");
C) fopen("bir_dosya.txt" , "w");
D) fopen("bir_dosya. txt" , "t");

DEĞERLENDİRME

Modül değerlendirmesinde verilen uygulamayı hatasız olarak tamamlayabildiyseniz bu modülü başarı ile tamamladınız demektir. Eğer, anlayamadığınız bir konu ya da bilgi eksikliğinden dolayı sonuca ulaşamadığınız bir nokta varsa bilgi sayfalarını tekrar okuyunuz.

Çözemediğiniz veya açıklık getiremediğiniz noktaları arkadaşlarınızla tartışmanız önerilmektedir. Arkadaşlarınızla uzlaşamadığınız noktaları öğretmeninize danışınız.

Ayrıca modül faaliyetleri ve araştırma çalışmaları sonunda kazandığınız bilgi ve becerilerin ölçülmesi için öğretmeniniz size ölçme araçları uygulayacaktır.

Ölçme sonuçlarına göre sizin modül ile ilgili durumunuz öğretmeniniz tarafından değerlendirilecektir.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	B
2	A
3	D
4	C
5	A
6	D
7	A
8	C
9	B
10	A

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	B
2	C
3	A
4	B
5	B
6	B
7	D
8	C
9	D
10	A

MODÜL DEĞERLENDİRME CEVAP ANAHTARI

SORU	CEVAP	SORU	CEVAP
1	B	11	C
2	A	12	B
3	C	13	A
4	B	14	D
5	D	15	D
6	B	16	A
7	D	17	D
8	C	18	D
9	B	19	B
10	D	20	C

ÖNERİLEN KAYNAKLAR

- ŞAMLI Mehmet, **PHP 5**, İstanbul, 2006.
- WELLING Luke, **PHP ve MYSQL Uzmanlar için**, İstanbul, 2006.
- http://programci.wordpress.com
- http://www.capraz.net
- http://www.kirbas.com
- http://www.e-hadi.net
- http://www.ceviz.net
- http://www.programlama.com
- http://www.php.com.tr

KAYNAKÇA

- DEMİRLİ Nihat, M.Yüksel İNAN, **Macromedia Dreamweaver MX 2004**, Ankara, 2005.
- ERDOĞAN Ahmet, **İnternet Programcılığı Notları**.
- OTANER Kayra, **PHP ve MySQL ile web Yazılım Geliştirme**, İstanbul, 2002.
- ŞAMLI Mehmet, **PHP 5**, İstanbul, 2006.
- WELLING Luke, **PHP ve MYSQL Uzmanlar için**, İstanbul, 2006.
- http://programci.wordpress.com
- http://www.capraz.net
- http://www.kirbas.com
- http://www.e-hadi.net
- http://www.ceviz.net
- http://www.programlama.com
- http://www.php.com.tr