فهرست مطالب

مقدمه

برای شروع نوشتن و اجرای برنامه PHP باید نرمافزار های xamp یا wamp را نصب کنیم. کد های برنامه را میتوان در هر محیط ویرایشگری نوشت در نهایت فایل پی اچ پی نوشته شده باید با پسوند php. ذخیره می‌شود.

دستور echo : برای چاپ یک متن در صفحه وب ازدستور echo استفاده می‌شود.البته این نکته مهم نباید فراموش شود که تمام دستورات PHP باید بین <?php?> قرار بگیرد در واقع php با ? < شروع و با ?> به پایان می‌رسد. همچنین برای پایان جمله ای که میخواهیم به چاپ برسانیم از (;) استفاده می‌کنیم. در PHP نوع و بزرگ و یا کوچک بودن فونت echo تاثیری در متن چاپی ما نخواهد داشت

<!DOCTYPE html> <html> <body> <?php ECHO "Hello World! "; echo "Hello World! "; EcHo "Hello World! "; ?> </body> </html>	خروجی Hello world! Hello world! Hello world!

توضیح اینکه < br > یک کد html است که هر گاه بخواهیم متن های چاپی ما در سطرهای جداگانه چاپ شوند از این کد استفاده می‌کنیم.

دستور print : دستور Print هم همانند دستور Echo وظیفه چاپ کدهای رشته‌ای را دارد و به همین دلیل کدها بین دو علامت " " قرار می‌گیرند و در انتها دستور به علامت ; به پایان می‌رسد.

print و echo هر دو یک عمل انجام می‌دهند اما echo از سرعت بالاتری برخوردار است .

```
<!DOCTYPE html>
<html>
<body>
// This is a single line comment
<?php
print "<h2>PHP is fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
</body>
</html>
```

خروجی دستور برابر است با

PHP is fun!

Hello world!

I'm about to learn PHP!

در راستای توضیح درباره PHP، اشاره ای به کد نویسی در html می‌کنیم. همان طور که در دستور Echo و print اشاره شد دستور echo برای چاپ یک متن استفاده می‌شد. حال اگر بخواهیم در صفحه وب سایت خود برای متن مورد نظرمان یک سر تیتر نیز بیان کنیم، باید یک سری کد html برای آن تعریف کنیم. شروع سر تیتر با <h1> و پایان آن با </h1> نمایش داده می‌شود،h ابتدای کلمه"header" در زبان انگلیسی است و عنوان مطلبی که می‌خواهیم درباره‌ی آن توضیح دهیم را نمایش می‌دهد.

توضیحات در PHP :

کامنت‌ها (توضیحات) در PHP، به صورت یک سری کد قبل از دستور echo نوشته می‌شود. این کدها در نرم‌افزار پی اچ پی هیچ تاثیری در نوشته ما ندارد و تنها کاربرد آن برای نویسنده است و اطلاعاتی را برای نویسنده برنامه که دوباره بعد از زمان از یک یا دوسال به آن رجوع می‌کند، یادآوری می‌کند. این کامنت مورد استفاده‌ی فردی که تازه برنامه را مشاهده می‌کند نیزمی‌باشد.

۱. قبل از شروع کامنت از علامت // استفاده کرده و در ادامه دیدگاه خود را در بیان می‌کنیم. این دیدگاه‌ها می‌تواند تاریخ، اطلاعاتی در باره‌ی متنی که در ادامه می‌خواهد چاپ شود و ... باشد.
۲. قبل از نوشتن کامنت از علامت # استفاده کرده و در ادامه دیدگاه خود را بیان کنید
۳. کامنت خود را بین دو علامت /* و*/ بیان کنید و تفاوت آن با دو مورد بالا این است که می‌تواند بیش از یک خط دیدگاه را در خود جای بدهد.

متغیرها در PHP : در پی اچ پی متغیرها را با علامت $ یا دلار نمایش می‌دهند.

۱. نام متغیرها تنها میتوانند حروف انگلیسی (a-z یا A-Z)، اعداد ۰-۹ و زیر خط (آندلاین) _ را شامل شود.
۲. نام متغیر هیچ گاه نمیتواند با عدد شروع شود.

```
<!DOCTYPE html>
<html>
<body>

<?php
$x=5;
$y=6;
$z=$x+$y;
echo $z;
?>

</body>
</html>
```

همان طور که در مثال بالا مشاهده می‌کنید سه متغیر y، x و z تعریف شده‌اند. هر یک از این متغیرها در PHP یک داده را معرفی می‌کند. متغیر z بین علامت‌های " " قرار نگرفته است، تنها دلیل این است که z یک متغیر رشته‌ای و یا در حقیقت متن نیست و جنس عدد است.

متغیرهای Local و Global

متغیرهای Local یا متغیرهایی محلی، جزو دسته ای از متغیرها در PHP هستند که باید در داخل تابع تعریف می‌شوند، در غیر این صورت در هیچ داده ای نمایش داده نمی‌شود و متغیرهای Global یا متغیرهای جهانی، متغیرهایی هستند که با تعریف آن‌ها در خارج از تابع می‌توانیم به آن‌ها سترسی داشته باشیم. در باره‌ی تابع یا Function در ادامه بیشتر توضیح می‌دهیم،

```
<!DOCTYPE html>
<html>
<body>
 <?php
$x=5; // global scope
 function myTest()
{
$y=10; // local scope
echo "Variable x is: $x";
echo "<br>";
echo "Variable y is: $y";
?>
 </body>
</html>
```

خروجی دستور

```
echo "<p>Test variables inside the function:<p>";
echo "Variable x is: $x";
echo "<br>";
echo "Variable y is: $y";
}
myTest();
 echo "<p>Test variables outside the function:<p>";
```

```
Test variables inside the function:

Variable x is:
Variable y is: 10

Test variables outside the function:

Variable x is: 5
Variable y is:
```

در مثال بالا x متغیر Global و y متغیر Local می‌باشد، یک بار متغیر محلی را برای چاپ صدا زده است و بار دیگر متغیر جهانی را برای چاپ صدا زده است. به همین دلیل در مورد اول در مقابل variable x is خالی است و در مورد دوم در مقابل Variable y is هیچ داده ای نمایش داده نشده است. در PHP می‌توانیم برای ذخیره متغیرهای جهانی از آرایه‌های $Global[index] استفاده کنیم. خروجی دستور زیر عدد ۱۵ است .

```
Global" lang="php" decode="true"
title="[  ]Global"]&lt;!DOCTYPE html&gt;
&lt;html&gt;
&lt;body&gt;

&lt;?php
$x=5;
$y=10;
```

```
function myTest()
{
$GLOBALS['y']=$GLOBALS['x']+$GLOBALS['y'];
}
myTest();
echo $y;
?&gt;

&lt;/body&gt;
&lt;/html&gt;
```

متغیر static :

هر گاه به متغیرها در PHP بخواهیم یک مقدار اولیه بدهیم و تنها یک بار این متغیر اولیه اجرا شود و در ادامه که آن را صدا میزنیم مقدار آخری که از تابع بدست آمده را جایگزین مقدار اولیه و بعد دستور را اجرا کند از متغیر static استفاده می‌کنیم.

```
<!DOCTYPE html>
<html>
<body>

<?php
function myTest()
{
static $x=0;
echo $x;
$x++;
}
myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
?>
</body>
</html>
```

خروجی

```
۰
۱
۲
۳
۴
```

داده‌ها در php : به انواع مختلف تقسیم می‌شوند:

۱- رشته‌ای، String ۲- اعداد صحیح، Integer ۳- اعداد ممیز شناور ، Floating point number

۴-بولین ،Boolean ۵- آرایه ،Array ۶- شی، Object ۷- تهی، Null

۱- رشته: یک سری کاراکتر است. یک رشته می‌تواند هر نوع متنی باشد. مانند: "Hello World!"

۲- عدد صحیح : تمام اعداد به غیر از اعداد اعشاری شامل اعداد صحیح می‌شوند.

در مثال زیر مدل‌های مختلف اعداد صحیح را مشاهده می‌کنیم. که برای نمایش نوع داده در خروجی از دستور var_dump($) استفاده می‌کنیم.

```
<html>
<body>
 <?php
$y = "Hello world!";
$x = 5985;
var_dump($x);
echo "<br>";
$x = -345; // negative number
var_dump($x);
echo "<br>";
$x = 0x8C; // hexadecimal number
var_dump($x);
echo "<br>";
$x = 047; // octal number
var_dump($x);
?>
</body>
</html>
```

خروجی دستور

```
int(5985)
int(-345)
int(140)
int(39)
```

در این خروجی دستور(var_dump باعث نمایش int که نوع داده است و همچنین عدد مقابل آن را نمایش می‌دهد.

۳- عدد با ممیز شناور می‌تواند به صورت اعشاری یا نمایی نوشته شود.

```
<!DOCTYPE html>
<html>
<body>

<?php
$x = 10.365;
var_dump($x);
echo "<br>";
$x = 2.4e3;
var_dump($x);
echo "<br>";
$x = 8E-5;
var_dump($x);
?>

</body>
</html>
```

در خروجی این مثال هم اعداد اعشاری می‌بینید و هم عدد نمایی

```
float(10.365)
float(2400)
float(8.0E-5)
```

۴- بولین : یک دستور منطقی است که تنها درست یا غلط بودن داده‌ها در PHP نشان می‌دهد

۵- داده‌های Array (آرایه) : آرایه‌ها چند مقدار را در یک متغیر نمایش می‌دهند. هنگامی که مقادیر ما بسیار زیاد است، استفاده از آرایه بهترین کزینه است.

```
<!DOCTYPE html>
<html>
<body>

<?php
$cars=array("Volvo","BMW","Toyota");
var_dump($cars);
?>

</body>
</html>
```

حال خروجی مثال بالا را بررسی کنیم.

array(3) { [0]=> string(5) "Volvo" [1]=> string(3) "BMW" [2]=> string(6) "Toyota" }

اگر به خروجی دقت کنید، مشاهده می‌کنید ابتدا دستور var_dump نوع داده‌ها در PHP مشخص و سپس دستور آرایه، تعداد آن را بیان کرده است، در ادامه نوع و تعداد داده‌های استفاده شده در آرایه یعنی Volvo، BMW، Toyota را نیز بیان می‌کند. در آرایه ها جایگاه عناصر از ۰ شمارش می‌شود، به همین دلیل درعلامت‌های [ ] اعداد صفر و سپس ۱ و ۲ را چاپ نموده است.

۶- داده‌های Object : یک شی نوعی داده است که اطلاعاتی را درباره چگونگی پردازش داده‌ها ذخیره می‌کند. در پی اچ پی، شی باید به صورت واضح بیان شده باشد. ابتدا ما باید یک کلاس برای یک شی اعلام کنیم، که با کلمه کلیدی class بیان می‌شود که شامل خواص، ویژگی‌ها و روش‌ها است. سپس با معرفی نوع داده در کلاس شی به استفاده از آن در مواقعی که می‌خواهیم، می‌پردازیم. در ادامه تدریس با مدل نوشتن class و Obj (آبجکت) بیشتر آشنا می‌شویم.

```
<?php
class Car
{
 var $color;
 function Car($color="green")
 {
  $this->color = $color;
 }
 function what_color()
 {
  return $this->color;
 }
}
?>
```

۷- داده‌های NULL : تهی نشان دهنده این است که آن داده‌ها درPHP ارزش ندارد و NULL تنها روش نشان دادن این است که آن مقدار ارزشی برابر با تهی دارد. این دستور برای تمایز بین رشته‌هایی که خالی هستند و ارزش‌های پوچ در پایگاه داده استفاده می‌شود.

```
<!DOCTYPE html>
<html>
<body>

<?php
$x="Hello world!";
$x=null;
var_dump($x);
?>

</body>
</html>
```

خروجی دستور

NULL

عملگرها در PHP

۱. حسابی ۲. انتسابی ۳. رشته‌ای ۴ . مقایسه‌ای ۵. منطقی ۶. آرایه

عملگر حسابی : عملگرحسابی یا ریاضی در پی اچ پی شامل عملیات جمع، تفریق، تقسیم، ضرب و قدر مطلق باقی‌مانده تقسیم است

نتیجه	مثال	نام	عملیات
جمع $x و $y	$x+$y	جمع	+
تفریق$x و $y	$x-$y	تفریق	-
ضرب $x و$y	$x*$y	ضرب	*
تقسیم $x و $y	$x/$y	تقسیم	/
باقیمانده تقسیم $x و $y	$x%$y	قدر مطلق	%

عملگرهای انتسابی : برای اختصاص دادن یک مقدار به یک متغیر از عملگر Assignment یا انتسابی استفاده می‌شود. پایه این عملیات در پی اچ پی، علامت "=" است.

توضیحات	مثال	انتساب
عملگر سمت چپ ارزشی برابر با آنچه که سمت راست بیان می‌شود، دارد	x=y	x=y
عمل جمع	x=x+y	x+=y
عمل تفریق	x=x-y	x-=y
عمل ضرب	x=x*y	x*=y
عمل تقسیم	x=X/y	X/=y
قدر مطلق	x=x%y	X%=y

عملگرهای رشته‌ای : این عملگرها در PHP به دو دسته تقسیم می‌شوند که هر دو وظیفه‌ی کنار هم قرار دادن متن‌ها و تشکیل یک جمله را دارند. این متغیرهای رشته ای هر کدام در یک خط به صورت جداگانه تعریف شده اند اما در خروجی نیازمند قرار گرفتن آن‌ها در کنار هم و به صورت یک جمله هستیم. در حقیقت با استفاده از (.) رشته های را در PHP ادغام میکنند.

عملگر	نام	مثال	نتیجه
.	الحاق	$txt1 = "Hello" $txt2 = $txt1 . world!"	متن ۲ شامل Hello world! می‌شود
=.	مامور الحاق	$txt1 = "Hello" $txt1 .= " world!"	متن۱شامل Hello world! می‌شود

هر دو عملگر یک خروجی را به همراه دارد اما یک تفاوت با هم دارند. برای عملگر اول باید دو تابع رشته‌ای با نام‌های جداگانه تعریف کنیم.

```
<!DOCTYPE html>
<html>
<body>
 <?php
$a = "Hello";
$b = $a . " world!";
echo $b; // outputs Hello world!
echo "<br>";
 $x="Hello";
$x .= " world!";
echo $x; // outputs Hello world!
?>
 </body>
</html>
```

خروجی دستور

Hello world!
Hello world!

در PHP میتوان رشته ها را با اعداد نیز ترکیب کرد:

```
$num=5;
$x="Test".$num;
```

عملگر کاهش یا افزایش : این عملگرها در PHP با افزایش یا کاهش یک واحد، در مقدار متغیرها، تغییر ایجاد می‌کنند. گاه به این صورت که مقدار متغیر را افزایش یا کاهش دهد و سپس نمایش دهد و یا همان مقدار را نمایش می دهد و در صورتی که متغیر را دوباره بخوانیم، افزایش یا کاهش اعمال شده را ارسال می‌کند. در جدول زیر با نماد این عملگرها و نوع کار آن‌ها بیشتر آشنا می‌شویم.

عملگر	نام	توضیحات
++$x	قبل از افزایش	افزایش x و سپس برگرداندن آن
$x++	ارسال افزایش	برگرداندن x و سپس افزایش x
- -$x	قبل از کاهش	کاهش y و سپس برگداندن آن
$x- -	ارسال کاهش	برگرداندن y و سپس کاهش آن

عملگرهای مقایسه ای : عملگرهای مقایسه‌ای در پی اچ پی، برای مقایسه ی دومقدار استفاده می‌شوند. این مقدارها می‌توانند از جنس عدد و متن (رشته ای) باشند.

عملگر	نام	مثال	نتیجه
==	مساوی	x == y	اگر x برابر با y باشد درست است
===	یکسان	x === y	اگر x برابر با y و از یک نوع باشد درست است
!=	نابرابر	x != y	اگر x با y برابر نباشد درست است
<>	نا برابر	y$ <> x$	اگر x با y برابر نباشد درست است
!==	غیر یکسان	x !== y	اگر x باy برابر یا از یک نوع نباشند درست است
>	بزرگتر	y$ < x$	اگر x از y بزرگتر باشد درست است
<	کوچک تر	y$ > x$	اگر x از y کوچک‌تر باشد درست است
>=	بزرگتر یا مساوی	y$ =< x$	اگر x بزرگ‌تر یا مساوی y باشد درست است
<=	کوچکتر یا مساوی	y$ => x$	اگر x کوچک‌تر یا مساوی y باشد درست است

عملگرهای منطقی : عملگرهای منطقی and، or، xor، &&، ||، !، در PHP بین دو متغیر قرارمی‌گیرند و شرایط درست یا نادرست بودن آن متغیرها را در PHP بیان می‌کند. لازم به ذکر است که عملگرهای && و || در اولویت بالاتری هستند. جدول زیر خروجی این متغیرها را در صورت قرار گرفتن عملگرهای پی اچ پی بین آن‌ها بیان می‌کند

عملگر	نام	مثال	نتیجه
and	و	$x and $y	اگر x و y درست باشد، درست است
Or	یا	$x or $y	اگر x یا y درست باشد، درست است
Xor	یا X	$x xor $y	اگر یکی از متغیرهای x یا y درست باشد درست است
&&	و	$x && $y	اگر x و y درست باشد، درست است
||	یا	$x || $y	اگر x یا y درست باشد، درست است
!	هیچ	!$x	اگر x درست نباشد، درست است

عملگرهای آرایه : عملگرهای آرایه در پی اچ پی برای مقایسه داده ها از جنس آرایه استفاده می‌شوند. در جدول زیر با نماد و نوع عملکرد این عملگرها در PHP آشنا می‌شوید.

عملگر	نام	مثال	نتیجه
+	اتصال	x + y	اتصال x و y
==	تساوی	x == y	اگر x و y یک مقدار داشته باشد، درست است
===	یکسان	x === y	اگر x وy دارای یک مقدار و از یک نوع باشند درست است.
=!	نا برابری	x != y	اگر x و y با هم برابر نباشند،درست است
<>	نا برابری	y$ <> x$	اگر x و y با هم برابر نباشند،درست است
==!	غیر یکسان	x !== y	اگر x با y یکسان نباشد، درست است

```
<!DOCTYPE html>
<html>
<body>
 <?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");
$z = $x + $y; // union of $x and $y
var_dump($z);
echo "<br>";
var_dump($x == $y);
echo "<br>";
var_dump($x === $y);
echo "<br>";
var_dump($x != $y);
echo "<br>";
var_dump($x <> $y);
echo "<br>";
var_dump($x !== $y);
?>
 </body>
<span style="font-family: 'Courier New',Verdana,Arial,Helvetica; font-size: small;"></html></span>
```

خروجی این دستور به صورت زیر است:

```
array(4) { ["a"]=> string(3) "red" ["b"]=> string(5) "green" ["c"]=> string(4) "blue"
["d"]=> string(6) "yellow" }
bool(false)
bool(false)
bool(true)
bool(true)
bool(true)
```

در دستور اول تنها خواسته شده است، آرایه های x و y در کنار هم نمایش داده شود. در دستور دوم به دلیل این که y و x یک مقدار را دارا نمی‌باشند نادرست اعلام شده است. در دستور سوم هم با این که متغیر x و y هر دو آرایه و ازجنس رشته ای هستند، اما چون مقادیر آن‌ها یکسان نیست، نادرست اعلام شده است. در دستور سوم، چهارم و پنجم به دلیل یکسان و برابر نبودن متغیر x و y درست اعلام شده است.

عبارت شرطی در PHP

بسیاری از اوقات هنگام نوشتن کدها در PHP می‌خواهید عملیات مختلفی در شرایط مختلف صورت پذیرد. در این زمان است که از عبارت شرطی در کد نویسی‌ها استفاده می‌کنیم.

عبارت شرطی If و عبارت if...esle

if (condition)
{
code to be executed if condition is true;
}
	if (condition)
	{
	code to be executed if condition is true;
	}
	else
	{
	code to be executed if condition is false;
	}

مثال

```
<!DOCTYPE html>
<html>
<body>
 <?php
$t=date("H");
if ($t<"20")
  {
  echo "Have a good day!";
  }
else
  {
  echo "Have a good night!";
  }
?>
 </body>
</html>
```

خروجی این دستور زمانی که ساعت کمتر از ۲۰ باشد !Have a good day را چاپ می‌کند و اگر ساعت بیشتر از ۲۰ را نشان بدهد Have good night را نمایش می‌دهد. پس در این جا بسته به ساعت دو جواب متفاوت می‌توانیم داشته باشیم.

عبارت if...elseif...else : همان طور که قبلا اشاره کردیم عبارت if...elseif...else، یعنی انتخاب یکی از چند بلوک کدها برای آنکه اجرا شود.همانند عبارات شرطی که قبلا توضیح دادیم، این عبارت شرطی را توضیح می‌دهیم.

```
if (condition)
  {
  code to be executed if condition is true;
  }
elseif (condition)
  {
  code to be executed if condition is true;
}
else
  {
  code to be executed if condition is false;
}
```

در این عبارت شرطی دو شرط تعریف می‌شود. به هر کدام از شرط ها یک else تعلق می‌گیرد تا درصورتی که شرط نادرست بودن کدهای دیگری اجرا شود. در یک مثال این قواعد را ببینیم.

عبارت شرطی switch : وقتی صحبت از عبارت switch می‌شود یعنی انتخاب یکی از بین بسیاری بلوک کدها برای آنکه اجرا شود. نحوه ی نوشتن این شرط با شرط ها دیگر کمی متفاوت است. برای نوشتن کدهایی خواناتر یا سرعت بیشتر عبارات شرطی بهتر است از قاعده switch در PHP استفاده کنیم. به نحوه نوشتن آن توجه کنید.

```
switch (n)
{
case label1:
  code to be executed if n=label1;
  break;
case label2:
  code to be executed if n=label2;
  break;
...
default:   code to be executed if n is
different from all labels;
}
```

در نوشتن این عبارت همانند مثال های قبل، پس از بیان شرط خود به صورت متغیر برای بیان موقعیت متغیرها شروع را با case و برای بستن موقعیت از break استفاده می‌کنیم. تمام موقعیت هایی که می‌خواهیم را می‌نویسیم هرگاه به هر کدام از آن ها رسید و شرط گفته شده برقرار بود آن را چاپ کرده و بقیه موقعیت ها را رها می‌کند و دستور به پایان می‌رسد. در حقیقت تنها یکی از این دستورها را که درست است را انتخاب می‌کند. و اگر هیچ یک از case ها درست نبود از کلمه default و بیان این که هیچ یک درست نیست می‌پردازیم.

حلقه‌ها در PHP

اغلب اوقات در نوشتن کدها نیاز است که چند دسته از کدها را بارها و بار ها اجرا کرد. برای اجرای این دستورها تنها کافی است از دستورهای حلقه استفاده کنید. انواع حلقه‌ها در PHP :

حلقه while و do...while : همان طور که در در بخش تعاریف اشاره کردیم، while به عنوان یک نوع از حلقه‌ها در PHP، دور حلقه خود را تا زمانی که شرط درست باشد ادامه می‌دهد. نحوه نوشتن این دسته از دستورات در پی اچ پی نیز همانند عبارت های شرطی در PHP، در مقابل while شرط را داخل ( ) قرار می‌دهیم و دستوری را که می‌خواهیم اجرا شود را می‌نویسیم. برخلاف حلقه while ابتدا کدها را در قسمت do انجام می‌دهد، سپس شرط را بررسی می‌کند.

```
while (condition is true)
  {
  code to be executed;
  }
```

```
do
  {
  code to be executed;
  }
while (condition is true);
```

حلقه for : همان طور که در بخش اول حلقه‌ها گفته شد، حلقه for کدها را در تعداد دفعات خاص اجرا می‌کند.پس زمانی این حلقه استفاده می‌شود که از تعداد دفعات اجرا اطلاع داشته باشید. در نحوه‌ی نوشتن این دستور سه عامل اهمیت دارد:

- مقدار اولیه ای که به دستور در حلقه می‌دهیم
- این دستور تا چه زمانی بررسی شود(مقدار نهایی)
- هر کدام از این مقدارها چگونه تغییر یابد(در هر دور چه مقدار به متغییر اضافه و یا از آن کم شود)

```
for (init counter; test counter; increment counter)
  {
  code to be executed;
  }
```

حلقه foreach : حلقه foreach در PHP تنها برای آرایه ها استفاده می‌شود. نحوه‌ی نوشتن این نوع حلقه‌ها در PHP به صورت زیر می‌باشد.

```
foreach ($array as $value)
  {
  code to be executed;
  }
```

در هر تکرار این حلقه، value(ارزش) به سراغ آرایه بعدی می‌رود و آن را بررسی می‌کند. با یک مثال بیشتر با این قواعد آشنا شوید.

```
<!DOCTYPE html>
<html>
<body>

<?php
$colors =
array("red","green","blue","yellow");
foreach ($colors as $value)
  {
  echo "$value <br>";
  }
?>
 </body>
</html>
```

خروجی این دستور

red
green
blue
yellow

تابع :

قطعه کدی که وظیفه خاصی را انجام می دهد. از ویژگی‌های توابع در PHP به موارد زیر می‌توان اشاره کرد:

- در کنار توابع موجود در پی اچ پی شما خودتان می‌توانید به صورت خلاقانه توابعی را ایجاد نمایید.
- این توابع می‌توانند به صورت پی در پی در برنامه اجرا شوند.

الیته این نکات را زمانی که خودتان می‌خواهید یک تابع جدید بنویسید در نظر بگیرید:

۱. نام تابع مورد نظر می‌تواند تنهاحروف و علامت‌ها را شامل شود. اعداد را در نام تابع قرار ندهید.
۲. نامش براساس نوع کاری که تابع انجام می‌دهد گذاشته شود.

نحوه‌ی نوشتن توابع درPHP به صورت زیر است:

```
function functionName()
{
code to be executed;
}
```

در مثال زیر تابع ای با نام(writeMsg) ایجاد کرده ایم. این تابع با استفاده از علامت { شروع و با علامت } بسته می‌شود. این مثال بسیار ساده است و تنها نحوه‌ی نوشتن تابع را در یک مثال نمایش می‌دهد. خروجی مثال زیر"Hello world!" ″است.

```
<!DOCTYPE html>
<html>
<body>
 <?php
function writeMsg()
{
echo "Hello world!";
}

writeMsg();
?>

</body>
</html>
```

پارامتر (Arguments): آرگومان‌ها در پی اچ پی همانند متغیرها هستند. اطلاعات از طریق این آرگومان‌ها به توابع منتقل می‌شود و نام آرگومان‌ها، مقابل نام تابع در بین () قرار می‌گیرد. در مثال زیر نام های مختلف مقابل اسم تابع قرار می‌گیرد اما همه آنها دارای یک نام خانوادگی هستند که این نام خانوادگی یک بار در دستور echo بیان شده است.

مقدار پیش فرض Argumentها : هنگامی که آرگومان را در تابع برابر با یک مقدار خاص قرار می‌دهیم، هنگامی که تابع را با همان مقدار خاص صدا می‌زنیم دیگر لازم نیست مقدار دوباره بنویسیم. فقط کافی است نام تابع را صدا بزنیم. برای درک بهتر این مطلب به مثال زیر دقت کنید.

```
<!DOCTYPE html>
<html>
<body>


<?php
function setHeight($minheight=50)
{
echo "The height is : $minheight <br>";
}

setHeight(350);
setHeight();
setHeight(135);
setHeight(80);
?>


</body>
</html>
```

خروجی این مثال:

The height is : 350
The height is : 50
The height is : 135
The height is : 80

مقادیر بازگشتی : این توابع مقادیر را با نوشتن عبارت return به تابع برمی‌گردانند. به مثال زیر توجه کنید.

```
<!DOCTYPE html>
<html>
<body>
 <?php
function sum($x,$y)
{
$z=$x+$y;
return $z;
}
 echo "5 + 10 = " . sum(5,10) . "<br>";
echo "7 + 13 = " . sum(7,13) . "<br>";
echo "2 + 4 = " . sum(2,4);
?>

</body>
</html>
```

خروجی این تابع

۵ + ۱۰ = ۱۵
۷ + ۱۳ = ۲۰
۲ + ۴ = ۶

آرایه :

آرایه ها متغیرهای خاصی هستند که می‌توانند چند ارزش را در یک زمان داشته باشند. در حقیقت یک متغیر می‌تواند چندین مقدار را در یک متغیر واحد ذخیره کند.

آرایه ها در PHP به سه دسته بر اساس نوع عملکردشان تقسیم می‌شوند:

- آرایه ایندکس شده یا عددی (Indexed arrays): آرایه‌هایی که با استفاده از عدد نمایش داده می‌شوند
- آرایه انجمنی(Associative arrays): آرایه‌هایی که با یک نام کلیدی فراخوان می‌شوند
- آرایه‌چند بعدی (Multidimensional arrays): آرایه‌هایی که شامل یک یا چند آرایه هستند

آرایه عددی : این نوع آرایه در اکثر زبان‌های کد نویسی وجو دارد. آرایه عددی در PHP را می‌توان به دو صورت نمایش داد، به صورت اتوماتیک خودش عدد بگیرد با توجه به این که می‌دانیم اعداد از ۰ در آرایه‌ها شروع می‌شود.

```
$cars=array("Volvo","BMW","Toyota");
```

طول آرایه- تابع ( )Count : در آرایه‌ها هرگاه بخواهیم تعداد عضوهای درون آرایه را بدانیم از تابع ( )count استفاده می‌کنیم. خروجی این تابع تنها یک عدد است که به تعداد متغیر ها اشاره دارد.

آرایه انجمنی یا(Associative Arrays) : آرایه هایی هستند که با یک نام کلیدی خاص خود فراخوان می‌شود. در شکل زیر هر یک از دایره ها عضوهای مقابل آرایه است که هر کدام با

نام کلیدی خاص خود (key name) فراخوان می‌شود.آرایه انجمنی در PHP را می‌توان به دو صورت نوشت. اولین روش این است که همه عضوهای آرایه را در یک خط در مقابل آرایه بنویسیم. مانند نمونه زیر:

```
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
```

و یا برای هر عضو آرایه به صورت جداگانه متغیر تعریف کنیم. در نمونه زیر به هر نام به صورت جداگانه متغیر $age اختصاص گرفته است.

```
$age['Peter']="35";
$age['Ben']="37";
$age['Joe']="43";
```

تفاوت آرایه انجمنی با آرایه عددی : آرایه انجمنی تنها نام مورد نظر را فراخوان می‌کند و دیگر لازم نیست که به شماره جایگاه آن عضو اشاره ای داشته باشد، اما در آرایه عددی تنها عضو ها با شماره جایگاهشان فراخوان می‌شود. یک نمونه مثال برای آرایه انجمنی در PHP مشاهده کنید.

```
<!DOCTYPE html>
<html>
<body>

<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");

foreach($age as $x=>$x_value)
  {
  echo "Key=" . $x . ", Value=" .
$x_value;
  echo "<br>";
  }
?>

</body>
</html>
```

آرایه دو بعدی : آرایه چند بعدی می‌تواند درون خود آرایه داشته باشد یعنی عناصر یک آرایه، خود یک آرایه است. در این راستا ما می‌توانیم آرایه دو بعدی و یا سه بعدی داشته باشیم. مثال زیر یک آرایه دو بعدی در PHP است.

```
<html>
<body>
 <?php
// A two-dimensional array
$cars = array
  (
  array("Volvo",100,96),
  array("BMW",60,59),
  array("Toyota",110,100)
  );
 echo $cars[0][0].": Ordered: ".$cars[0][1].". Sold: ".$cars[0][2]."<br>";
echo $cars[1][0].": Ordered: ".$cars[1][1].". Sold: ".$cars[1][2]."<br>";
echo $cars[2][0].": Ordered: ".$cars[2][1].". Sold: ".$cars[2][2]."<br>";
?>
 </body>
</html>
```

آرایه چند بعدی : می‌تواند خود شامل آرایه باشد و هر یک از آرایه خود شامل چندین آرایه است.

```
$families = array
  (
  "Griffin"=>array
  (
  "Peter",
  "Lois",
  "Megan"
  ),
  "Quagmire"=>array
  (
  "Glenn"
  ),
  "Brown"=>array
  (
  "Cleveland",
  "Loretta",
  "Junior"
  )
  );
```

خروجی

```
Array
(
[Griffin] => Array
(
[0] => Peter
[1] => Lois
[2] => Megan
)
[Quagmire] => Array
```

ادامه خروجی

```
(
[0] => Glenn
)
[Brown] => Array
(
[0] => Cleveland
[1] => Loretta
[2] => Junior
)
)
```

متغیرهای سراسری جهانی (GLOBAL)

متغیرهای "بسیار جهانی" اولین بار در نسخه ۴.۱ پی اچ پی ارائه شد. این متغیرها برای در دسترس بودن در تمام بخش ها ساخته شده اند. چندین متغیر superglobal از پیش تعریف شده در PHP، متغیر های معرفی شده در زیر هستند بدین معنی که بدون در نظر گرفتن بخش، تابع و کلاس و یا هر فایلی قابل استفاده هستند.

۱- $GLOBALS ۲- $_SERVER ۳- $-REQUEST ۴- $_POST ۵- $_GET

۶- $_FILES ۷- $_ENV ۸- $_COOKIE ۹- $_SESSION

متغیر $GLOBALS : متغیر $GLOBALS از جمله متغیر های سوپر گلوبال است که برای دسترسی به تمام متغیرهای سوپر گلوبال هر جای اسکریپت ها در PHP نوشته می‌شود. در PHP تمام متغیر های جهانی در متغیر $GLOBALS ذخیره می‌شود.

```
<!DOCTYPE html>
<html>
<body>

<?php
$x = 75;
$y = 25;

function addition()
{
$GLOBALS['z'] = $GLOBALS['x'] +
$GLOBALS['y'];
}
 addition();
echo $z;
?>
 </body>
</html>
```

در این دستور یک تابع جمع و درون تابع جمع، متغیر $GLOBALS تعریف شده است. همان طور که گفته شد،متغیر $GLOBALS تمام متغیر ها را در هر جای PHP می‌خواند. در این مثال متغیرهای X وY را خوانده است تا با توسط تابع جمع حاصلشان بدست آید. چون Z با استفاده از متغیر GLOBALS تعریف شده است خارج از تابع هم می‌توان به آن دسترسی داشت.خروجی این مثال عدد ۱۰۰ است.

متغیر $_SERVER : متغیر $_SERVER از جمله متغیر های سوپر گلوبال در PHP است که اطلاعاتی درباره عنوان ها، مسیر ها و محل دستورها می‌دهد.

```
<html>
<body>

<?php
echo $_SERVER['PHP_SELF'];
echo "<br>";
echo $_SERVER['SERVER_NAME'];
echo "<br>";
echo $_SERVER['HTTP_HOST'];
echo "<br>";
echo $_SERVER['HTTP_REFERER'];
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];
?>
 </body>
</html>
```

با توجه به این که این مثال برگرفته از سایت w3schools استف خروجی باید ویژگی های خواسته شده باید برای این سایت باشد. در این مثال با استفاده از متغیر گلوبال$-SERVER اطلاعی از قبیل نام سرور، نام دستورها، http مراجعه کنندگان، هاست و... را نشان چاپ می‌کند. خروجی مثال :

/php/demo_global_server.php

www.w3schools.com

www.w3schools.com

http://www.w3schools.com/php/showphp.asp?filename=demo_global_server

Mozilla/5.0 (Windows NT 6.1; rv:23.0) Gecko/20100101 Firefox/23.0 AlexaToolbar/alxf-2.19 AlexaToolbar/pGURBh8f-2.2

/php/demo_global_server.php

متغیر $_REQUEST : متغیر $_REQUST یکی از متغیر های گلوبال که برای فرم های موجود در سایت ها استفاده می‌شود. این متغیر اطلاعات یک فرم را که توسط کاربر نوشته شده است جمع آوری می‌کند.

مثال زیر یک فرم را با یک سری فیلد های ورودی و دکمه تایید نشان می‌دهد که کاربر پس از تایید اطلاعات وارد شده، اطلاعات با برچسب <form> ذخیره می‌شود.

```
<!DOCTYPE html>
<html>
<body>
 <form method="post"
action="<?php echo
$_SERVER['PHP_SELF'];?>">
```

```
<?php
$name = $_REQUEST['fname'];
echo $name;
?>
```

```
Name: <input type="text"
name="fname">
<input type="submit">
</form>
</body>
</html>
```

در این مثال خروجی با توجه به آن چه که در فرم وارد می‌کنید متفاوت است و php تنها این اطلاعات را جمع آوری می‌کند و به چاپ می‌رساند.

متغیر $_POST : متغیر سوپر گلوبال $_POST ویژگی ای همانند متغیر $_REQUEST دارد با این تفاوت که با روش نوشتاری "method="post استفاده می‌شود و به دلیل عبور دادن متغیرها استفاده زیادی دارد. مثال زیر همانند مثال قبل است با این تفاوت که در بخش echo از متغیر $_post استفاده شده است.

```
<!DOCTYPE html>
<html>
<body>

<form method="post" action="<?php echo
$_SERVER['PHP_SELF'];?>">
Name: <input type="text" name="fname">
<input type="submit">
</form>

<?php
$name = $_POST['fname'];
echo $name;
?>

</body>
</html>
```

خروجی این مثال با توجه به داده ای وارد فرم می‌شود متفاوت است.

متغیر $_GET : متغیر $_GET همانند متغیرهای $_REQUST و $_POST داده های را در یک فرم پس از ارسال جمع آوری می‌کند با این تفاوت که برای استفاده از این متغیر باید از روش نوشتاری "method="get استفاده شود. همچنین متغیر $_get داده هایی که در URL ها هم فرستاده شده است را می‌تواند ذخیره کند.

فرض کنید که یک صفحه html که حاوی یک لینک ها با پارامتر ها است.

```
<html>
<body>

<a href="test_get.php?subject=PHP&web=W3schools.com">Test $GET</a>

</body>
</html>
```

هنگامی که یک کابر بر روی لینک "Test $GET" پارامترهای موضوع و وب به آدرس "test_get.php" فرستاده می‌شود و شما می‌تواید با متغیر $_GET به اطلاعات "test_get.php" دست پیدا کنید

```
<!DOCTYPE html>
<html>
<body>
```

```
<a href="test_get.php?subject=PHP&web=W3schools.com">Test $GET</a>
 </body>
</html>
```

خروجی دستور زیر با توجه به متغیر $_Get یک لینک است که موضوع و نام وب را بیان می‌کند.

Test $GET

فایل: برای فایل در PHP توابع مختلفی تعریف می‌شود که برای باز کردن، بستن، خواندن یک فایل استفاده می‌شود. به اختصار با مهمترین این توابع برای فایل آشنا می‌شویم.

تابع fopen() : برای باز کردن فایل ها در پی اچ پی استفاده می‌شود. اولین پارامتر این تابع مشخص می‌کند که تابع با چه نامی باید باز شود و دومین پارامتر مدل فایل را مشخص میکند.

تابع fclose() : این تابع برای بستن فایل در PHP استفاده می‌شود

فایل ها ممکن است با یکی از مدل های زیر باز شوند:

مدل	توضیح
r	تنها می‌خواند. در ابتدای فایل شروع می‌شود
r+	می‌خواند/ می‌نویسد. در ابتدای فایل شروع می‌شود

w	تنها می‌نویسد. یک فایل را باز کرده و متن داخلش را پاک می‌کند و یا یک فایل جدیدی که وجود ندارد را می‌سازد
w+	می‌خواند و می‌نویسد. یک فایل را باز کرده و متن داخلش را پاک می‌کند و یا یک فایل جدیدی که وجود ندارد را می‌سازد
a	اضافه کردن. باز می‌کند و در انتهای فایل می‌نویسد یا یک فایل جدید که وجود ندارد را ایجاد می‌کند
a+	می‌خواند و اضافه می‌کند. محتویات فایل را با نوشتن آن در آخر فایل حفظ می‌کند.
x	تنها می‌نویسد.یک فایل جدید می‌سازد. اگر در فایل اشتباه و خطا وجود دارد برمیگرداند.
x+	می‌خواند و می‌نویسد. یک فایل جدید می‌سازد. اگر در فایل اشتباه و خطا وجود دارد برمیگرداند

- نکته: اگر تابع fopen() نتوانستد فایل مشخصی را باز کند تنها ۰ را بر می‌گرداند.

مثال زیر پیامی را در صورتی که فایل مشخصی را نتواند باز کند را ایجاد می‌کند.

```
<html>
<body>
 <?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
fclose($file);
?>
 </body>
</html>
```

در این مثال اگر فایل welcome.txt باز نشود پیام نمی‌تواند فایلی را باز کند خارج می‌شود.

تابع feof() : اگر انتهای فایل در دسترس باشد، آن را چک می‌کند. این تابع برای حلقه هایی که تعداد طول داده های آن مشخص نیست بسیار مناسب است.

نحوه نوشتن این فایل به صورت زیر است: `if (feof($file)) echo "End of file";`

تابع fgets() : این تابع برای خواندن تک تک خط ها در یک فایل استفاده می‌شود.

مثال زیر هر خط فایل خوانده می‌شود تا زمانی که انتهای فایل دسترسی پیدا کند.

```
<?php
$file = fopen("welcome.txt", "r") or exit("Unable to open file!");
//Output a line of the file until the end is reached
while(!feof($file))
  {
  echo fgets($file). "<br>";
  }
fclose($file);
?>
```

تابع fgetc() : این تابع تک تک کاراکترهای یک فایل را می‌خواند. به این صورت که پس از خواند فایل اشاره گر آن را به کاراکتر بعدی انتقال می‌دهد.

مثال زیر تک تک کاراکتر های یک فایل را می‌خواند تا زمانی که به انتهای فایل دسترسی پیدا کند.

```
<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
while (!feof($file))
  {
  echo fgetc($file);
  }
fclose($file);
?>
```

تابع date : برای قالب بندی تاریخ/زمان، محاسبه زمان، نمایش یک تاریخ خاص، ایجاد یک برچسب زمانی و... استفاده می‌شود. date(format,timestamp)

در بخش format باید مقداری نوشته شده باشد. قالب بندی را برای برچسب زمان مشخص می‌کند. پر کردن بخش timestamp اختیاری است. برچسب زمان را مشخص می‌کند. به طور پیش فرض تاریخ و زمان آن لحظه است.پارامترهای مورد نیاز بخش format در تابع date() مشخص می‌کند که چگونه تاریخ و زمان را قالب بندی کنید.

کاراکترهایی که می‌توانیم استفاده کنیم به صورت زیر است:

- d : نشان دهنده روزهای یک ماه(۰-۳۱)
- m : نشان دهنده یک ماه (۰-۱۲)
- Y : نشان دهنده یک سال (با چهار رقم)

کاراکترهای دیگر مانند "/"، "."، "-" در بین حروف قالب بندی قرار داده شوند، به صورت زیر:

خروجی دستور	<?php echo date("Y/m/d") . " "; echo date("Y.m.d") . " "; echo date("Y-m-d"); ?>
۲۰۰۹/۰۵/۱۱ ۲۰۰۹.۰۵.۱۱ ۲۰۰۹-۰۵-۱۱	

تابع mktime() : برای برگرداندن یک برچسب زمان unix استفاده می‌شود. طرز نوشتن این تابع به صورت زیر است: mktime(hour,minute,second,month,day,year,is_dst)

خروجی	<?php $tomorrow = mktime(0,0,0,date("m"),date("d")+1,date("Y")); echo "Tomorrow is ".date("Y/m/d", $tomorrow); ?>
Tomorrow is 2009/05/12	

برنامه نویسی شی گرا

در دنیای واقعی یک سری شی در اطراف ما است که هر کدام کار انجام میدهند برای مثال : ماشین و لامپ. با استفاده از ماشین میتوانیم رانندگی کنیم و لامپ ایجاد روشنایی میکند. این ها تنها عملکردی است که ما به چشم می بینیم اما این اشیا فقط این ویژگی ها را دارا نمی باشند. ماشین میتواند با یک سرعت و مسیر خاص حرکت کند، لامپ میتواند روشن یا خاموش باشد، گرما تولید و در یک زمان مشخص میزان وات مشخصی مصرف کند.

در PHP نیز اشیا همان گونه هستند، میتوانید به آنها ویژگی خاص اختصاص دهید با این تفاوت که کاربران این ویژگی را در صفحه اصلی سایت مشاهده و با آن ها ارتباط برقرا میکنند. شما میتوانید با استفاده از برنامه نویسی شی گرا به هر یک از این اشیا بگویید چه صفتی و ویژ گی داشته باشند، چگونه ارزیابی شوند و یا تغییر کنند.

PHP یک راه بسیار ساده برای استفاده از اشیا در برنامه نویسی شی گرا تعریف کرده است و آن هم کلاس ها هستند. کلاس ها یک بسته بندی مشخص از اشیا با تمامی ویژگی ها و روش ها است. میتوانیم این گونه تصور کنید که کلاس ها نماینده های برنامه ریزی کننده برای اشیا هستند و رابط آن ها با افراد در محیط اصلی سایت کد ها PHP است! میتوانید یک بار کلاس PHP را تعریف کنید و در صورتی که نیاز داشتید بی نهایت بار از آن استفاده کنید.

یک نمونه از این برنامه نویسی شی گر

```
<?php
class Image {
    public function Image() {
        echo "We just created and object!";
    }
}
$image = new Image(); // prints "We just created and object!"
?>
```

کلاس ها در PHP : برای تعریف یک کلاس ها در PHP از کلمه class استفاده میکنیم. این کلاس میتواند شامل خصوصیات مختلف عمومی یا خصوصی باشد. همیشه کلاس ها با یک نام مشخص میشوند که بیان کننده هدف ما از طراحی است. در زمان نام گذاری یک کلاس دقت داشته باشید که نام یک کلاس نمی تواند عدد باشد.

پس از معرفی کلاس ها به معرفی خصوصیات می پردازیم. خاصیت ها در واقع متغیرهایی هستند که درون کلاس تعریف میشوند. خصوصیات در کلاس ها را با استفاده از مثال زیر توضیح خواهم داد.

```
<?php
class Emailer {
private $sender;
private $recipients;
private $subject;
private $body;
```

```
public function __construct($sender) {
$this->sender = $sender;
$this->recipients = array();
}
public function AddRecipients($recipient) {
array_push($this->recipients, $recipient);
}
public function SetSubject($subject) {
$this->subject = $subject;
}
public function SetBody($body) {
$this->body = $body;
}
public function SendEmail() {
foreach ($this->recipients as $recipient) {
$result = mail($recipient, $this->subject, $this->body, "From: {$this->sender}\r\n");
if ($result) {
echo "Mail successfully sent to {$recipient}<br/>" . PHP_EOL;
}
}
}
}
?>
```

نام این کلاس Emailer است. همان طور که مشاهده می کنید خصوصیات درون کلاس ها به صورت private مشخص شده اند. private بودن خواص یعنی تنها میتوانید درون کلاس از آن ها استفاده کنید بنابراین باعث افزایش امنیت در برنامه نویسی تعریف اشیا میشود. در برخی کتاب ها یا منبع های دیگر شاید خاصیت ها را با نام "فیلد" نیز مشاهده کنید. در این مثال sender، $recipients، $subject و body خصوصیات کلاس Emailer هستند.

عنصر بعدی که با آن در کلاس های PHP مواجه میشوید، متد ها هستند. متد ها نیز چیز جدیدی نیستند، متد ها همان توابعی هستند که درون کلاس ها در PHP تعریف میشوند. در این مثال ۵ متد داریم. همان طور که مشاهده میکنید متد ها را به صورت public یا عمومی تعریف کرده ایم.

یعنی هرگاه در برنامه یک شی ایجاد کنیم، متد ها از طریق شی های مرتبط میتوانند فراخوانی شوند. به عبارت دیگر در صورتی که خصوصی با private باشد تنها درون کلا در دسترس است و از طریق اشیا نمی توانیم به آن متد دسترسی داشته باشیم.

نام گذاری عناصر درون کلاس : برای نام گذاری عناصر درون کلاس ها یک قرار داد جهانی وجود دارد. همان طور که گفتم این تنها قرار داد است نه قانون.

- حرف اول عناصر Private کوچک و حروف اول دیگر کلمات درون این عنصر بزرگ باشد مانند myFirstValue:
- حرف اول تمامی کلمات عناصر public بزرگ و بقیه حروف به صورت کوچک باشد مانند : MyFirstValue

فرم درHTML

مثال زیر یک نمونه فرم در HTML با دو ورودی و یک دکمه برای ارسال اطلاعات است.

```
<!DOCTYPE HTML>
<html>
<body>

<form action="welcome.php"
method="post">
Name: <input type="text"
name="name"><br>
E-mail: <input type="text"
name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

در این دستور یک فرم با روش post نوشته شده است. و نام و ایمیل را از کاربر می‌خواهد

Post برای فرم HTML : وقتی کاربر این بخش ها را با اطلاعات خود پر می‌کند داده ها توسط روش post ذخیره شده و در فایلی به نام welcome.php بررسی می‌شود. (نام این فایل از دستور نوشته شده توسط html می‌آید) و در نهایت با دستور php زیر به نمایش در می‌آید.

```
<html>
<body>
```

```
 Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>
 </body>
</html>
```

در این دستور نام کاربر که در فرم خواسته شده بود و ایمیل او چاپ می شود.

welcome Baharan

Your email address is baharan.doe@example.com

همین نتیجه را می‌توان با استفاده روش GET در HTML بدست آورد. این دستور را با دستور نوشتن فرم در HTML با استفاده از POST مقایسه کنید.

```
<!DOCTYPE HTML>
<html>
<body>
 <form action="welcome_get.php" method="get">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>
 </body>
</html>
```

خروجی این دستور هم همانند دستور اول شامل یک مکان برای نوشتن فایل و یک مکان برای نوشتن ایمیل کاربر و دکمه ارسال است.

دستور Get در PHP برای فرم HTML : وقتی کاربر اطلاعات خود را وارد می‌کند، تمام اطلاعات با استفاده از GET در فایل Welcom-get.php ذخیره می‌شود. این نام را در دستور html تعریف کرده بودیم. و خروجی نام و ایمیل کاربر را در اختیارتان قرار می‌دهد.

<html> <body> Welcome <?php echo $_GET["name"]; ?> Your email address is: <?php echo $_GET["email"]; ?> </body> </html>	خروجی Welcome Baharan Your email address is baharan.doe@example.com

فقط به این نکته توجه داشته باشید که ما در این جا تنها به کد هایی برای نوشتن یک فرم اشاره کردیم اما این کد ها حفاظتی از کد های شما نمی‌کند. برای حفاظت نیاز به دستورات دیگری هست که در ادامه آموزش به آنها نیز اشاره خواهیم کرد.

مقایسه POST و GET

GET و POST هر دو می‌توانند یک آرایه را ایجاد کنند. این آرایه ها دارای کلید و ارزش است که درآن کلید نام کنترل ها و ارزش ، داده های ورودی در فرم است که کاربر وارد می‌کند. هر دوی $_GET و $_POST جزو متغیر های سوپر گلوبال هستند و بدون در نظر گرفتن دامنه و کلاس یا تابع همیشه در دسترس هستند.

$_GET آرایه ای از متغیرها است که از طریق پارامتر های URL به اسکریپت های فعلی منتقل می‌شوند. اطلاعاتی که توسط $_GET نوشته می‌شود برای همگان قابل مشاهده است و در میزان اطلاعاتی که از فرد می‌خواهید بگیرید دارای محدودیت است. (بیش از ۲۰۰۰ کاراکتر نمیتوان در آن نوشت) به همین دلیل از GET نباید برای گرفتن کلمه عبور و اطلاعات مهم استفاده کرد.

$-POST آرایه ای از متغیرها است که از طریق روش HTTP POST به اسکریپت های فعلی منتقل می‌شوند. اطلاعاتی که در $_POST نوشته می‌شود برخلاف $_GET، برای همگان قابل رویت نیست و در میزان اطلاعاتی که می‌توانیم در آن قرار دهیم هیچ محدودیتی نخواهیم داشت. اما چون متغیر ها در این روش در URL ها نمایش داده نمی شوند، نمی توان آن صفحه را برای کاربر نشانه گذاری کرد

اعتبارسنجی فرم PHP

اعتبارسنجی فرم PHP مهم ترین موضوع برای طراحان سایت است. بررسی این موضوع که کاربران داده هایی که در یک فرم وارد می‌کنند درست است یا خیر یا به محدودیت های

گذاشته شده در بخش های مختلف فرم توجه کرده اند از مهمترین نیاز های طراحان سایت در یک فرم است. این اطلاعات می‌تواند داده هایی که کاربر وارد میکند، کوکی هایی که فرستاده میشوند، داده های سرویس های وب را شامل باشد.

فیلد های متنی: بخش های این فرم که در آن المان هایی از جنس متن قرار می‌گیرند را معرفی میکند. این بخش ها شامل نام، ایمیل، وب سایت و بخش توضیحات است. کد HTML این بخش ها به صورت زیر تعریف شده اند

```
Name: <input type="text" name="name">
E-mail: <input type="text" name="email">
Website: <input type="text" name="website">
Comment: <textarea name="comment" rows="5" cols="40"></textarea>
```

این کد ها داده های ورودی بخش نام، ایمیل، وب سایت و توضیحات را دریافت می‌کنند تا در ادامه داده های ورودی را بررسی شوند به صورت خروجی درآیند.

کلیدهای رادیویی : در این فرم برای قسمت جنسیت افراد کلیدهای رادیویی قرار داده شده تا با انتخاب یکی از آنها جنسیت کاربر مشخص شود. از این کلید های رادیویی در هر جایی که نیاز به انتخاب گزینه ها باشد می‌توان استفاده کرد. نوشتن این کد ها در HTML به صورت زیر است:

```
Gender:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
```

عناصر فرم : برای قرار دادن این ورودی ها به صورت فرم از کد زیر استفاده می‌کنیم. این کد شامل بخش های مختلفی است که در ادامه آن ها را به تفصیل توضیح می‌دهیم:

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

وقتی که فرم تایید شود، داده ها به بخش method=post فرستاده می‌شوند.

نکته مهم در فرم های PHP

نکته ۱ : $_SERVER["PHP_SELF"] یک متغیر سوپر گلوبال است که نام فایل در حال اجرا را باز می‌گرداند و در همان صفحه به نمایش می‌گذارد به جای آن که در صفحه دیگری اطلاعات را نشان دهد.

نکته ۲: تابع htmlspecialchars() کاراکتر های مورد نظر در PHP را به یکسری کارکتر های خاص در HTML تبدیل می‌کند. این تبدیل از سو استفاده افراد در فرم مورد نظر جلوگیری میکند. برای مثال کاراکتر<and> را در HTML به صورت gt& lt; and&; نمایش میدهد.

نکته ۳: تابع $_SERVER["PHP_SELF"] جزو متغیرهایی است که می‌تواند مورد استفاده هکرها قرار گیرد. یک کاربر میتواند با اضافه کردن یک اسلش "/" به دامنه شما کد تزریق وارد میکنند و سایت شما را هک کند. (کد تزریق یا xss از روش هایی است که مورد استفاده هکر ها است. برای جبران این ضعف php راه حلی قرار داده است. ابتدا ببینیم که تابع $_SERVER["PHP_SELF"] چگونه مورد استفاده هکرها قرار میگیرد.

فرض کنید ما یک فرم به نام test_form.php داشته باشیم:

```
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
```

حال، اگر کاربر یک بخشی مانند "http://www.example.com/test_form.php" به آدرس بار اضافه کند، به صورت زیر ترجمه میشود:

```
<form method="post" action="test_form.php">
```

حال اگر یک کاربر آدرسی مبتنی بر هک شدن را در آدرس بار بنویسد.

```
http://www.example.com/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E
```

این آدرس به صورت زیر ترجمه میکند:

```
<form method="post" action="test_form.php"/><script>alert('hacked')</script>
```

این کد یک برچسب اسکریپت و یک متن اخطار اضافه میکند. هنگامی که فردی وارد آن صفحه شود کد جاوا اسکریپت اجرا میشود و آن فرد اخطار هک شدن را مشاهده میکند. این روش آسان ترین روش برای هکر ها در استفاده از تابع $_SERVER["PHP_SELF"] است.

نکته ۴: با استفاده از تابع htmlspecialchars() میتوان از هک شدن
تابع $_SERVER["PHP_SELF"] جلوگیری کرد. نوشتن این کد برای فرم ها در php به صورت زیر است:

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

همان طور که گفته شد تابع htmlspecialchars() کارکترها را در html تغییر میدهد بنابراین هکر هنگامی که کدی را تزریق میکند آن کد به صورت زیر نمایش داده میشود:

```
<form method="post"
action="test_form.php/&amp;quot;&amp;gt;&amp;lt;script&amp;gt;alert('hacked')&amp;lt;/script&amp;gt;">
```

بنابراین هک شدن ناموفق میشود و امنیت فرم بالا رفته و هیچ خطری فرم را تهدید نخواهد کرد.

بررسی داده های فرم با PHP

اولین قدم فرستادن داده ها در PHP به تابع htmlspecialchars() است. با نوشتن تابع htmlspecialchars() هنگامی که کاربری قصد نوشتن اسریپت <script>location.href('http://www.hacked.com')</script> را داشته باشد، این کد به صورت زیر نمایش داده میشود و باعث ایجاد امنیت در صفحه و ایمیل میشود:

<script>location.href('http://www.hacked.com');</script>

همچنین از دو دستور دیگر برای ارسال یک فرم استفاده میکنیم:

تابع trim() : با استفاده از این تابع میتوان فاصله های اضافی گذاشته شده توسط کاربر را در فرم و یا خط های اضافی را پاک کرد.

تابع stripslashes() : با استفاده از این تابع در PHP میتوان / "های گذاشته شده توسط کاربر را حذف کرد.

قدم بعدی ایجاد کردن تابعی است که بتوان داده های نوشته شده در فرم را برای ما بررسی کند. نام این تابع را test_input() قرار میدهیم. حال میتوان تمام متغیرهای نوشته شده در $_POST() را با استفاده از تابع test_input () بررسی کرد.

```
<!DOCTYPE HTML>
<html>
<head>
</head>
<body>
 <?php
// define variables and set to empty values
$name = $email = $gender = $comment = $website = "";
 if ($_SERVER["REQUEST_METHOD"] == "POST")
{
   $name = test_input($_POST["name"]);
   $email = test_input($_POST["email"]);
   $website = test_input($_POST["website"]);
   $comment = test_input($_POST["comment"]);
   $gender = test_input($_POST["gender"]);
}
 function test_input($data)
{
   $data = trim($data);
   $data = stripslashes($data);
   $data = htmlspecialchars($data);
   return $data;
}
?>
 <h2>PHP Form Validation Example</h2>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
   Name: <input type="text" name="name">
   <br><br>
```

```
  E-mail: <input type="text" name="email">
  <br><br>
  Website: <input type="text" name="website">
  <br><br>
  Comment: <textarea name="comment" rows="5" cols="40"></textarea>
  <br><br>
  Gender:
  <input type="radio" name="gender" value="female">Female
  <input type="radio" name="gender" value="male">Male
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>
 <?php
echo "<h2>Your Input:</h2>";
echo $name;
echo "<br>";
echo $email;
echo "<br>";
echo $website;
echo "<br>";
echo $comment;
echo "<br>";
echo $gender;
?>
 </body>
</html>'
```

توجه داشته باشید انچه که در ابتدا نوشته و ارسال خواهد شد توسط تابع SERVER["REQUEST_METHOD_$e"] بررسی میشود. اگر اعتبارسنجی فرم PHP مورد تایید و داد ها از جنس متغییر POST بودند آن را نمایش میدهد. اگر این شرایط برقرار نبود صفحه دوباره لود شده و یک فرم خالی نمایش داده میشود.

تکمیل اعتبارسنجی فرم PHP : در این بخش میخواهیم برای افزایش اعتبارسنجی فرم PHP پر کردن بخش های نام، ایمیل، جنسیت را اجباری کنیم. در صورت خالی بودن این بخش ها خطا نشان بدهد و پر کردن بخش های وبسایت و کامنت را اختیاری باشد.

بخش	شرایط
نام	نوشتن نام لازم است
ایمیل	نوشتن یک ایمیل معتبر لازم است
وبسایت	در صورت داشتن وبسایت URL آن را نوشته شود
کامنت	در صورت داشتن توضیحات، متنی نوشته شود
جنسیت	یکی از گزینه های را انتخاب کنید

در این فرم اجباری بودن هر بخش با * نشان داده شده است. در ادامه یک سری کد نمایش داده شده است که در آن متغیرهای $nameErr, $emailErr, $genderErr و $websiteErr تعریف شده اند. این متغیرها پیغام خطا را در خود ذخیره میکنند. همچنین یک بخش if else اضافه میکنیم. هنگام بررسی متغیر $_POST درون آن خالی باشد پیغام خطا نمایش داده میشود و در صورتی که درون این متغیر خالی نباشد، داده ها برای بررسی به تابع test_input() ارجاع داده میشوند. این کدها به صورت زیر نمایش داده میشوند.

```
<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";
 if ($_SERVER["REQUEST_METHOD"] == "POST")
{
   if (empty($_POST["name"]))
    {$nameErr = "Name is required";}
  else
    {$name = test_input($_POST["name"]);}

  if (empty($_POST["email"]))
    {$emailErr = "Email is required";}
  else
```

```
  {$email = test_input($_POST["email"]);}

 if (empty($_POST["website"]))
  {$website = "";}
 else
  {$website = test_input($_POST["website"]);}

 if (empty($_POST["comment"]))
  {$comment = "";}
 else
  {$comment = test_input($_POST["comment"]);}

 if (empty($_POST["gender"]))
  {$genderErr = "Gender is required";}
 else
  {$gender = test_input($_POST["gender"]);}
}
?>
```

حال برای این که یکی از بخش های نام، ایمیل، یا جنسیت خالی بود به کاربر نشان دهد که این فیلد باید پر شود یک متنی در مقابل ان به عنوان اخطار نشان میدهد. برای نوشتن این اخطار از دستور span در HTML استفاده میکنیم که این اخطار را در کنار آن بخش خالی نمایش دهد. در این span یک دستور echo نوشته میشود که آنچه که در متغیرهای $genderErr ,emailErr و $websiteErr به عنوان متن خطا نوشته شده بود، نمایش دهد.

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
 Name: <input type="text" name="name">
<span class="error">* <?php echo $nameErr;?></span>
<br><br>
E-mail:
<input type="text" name="email">
<span class="error">* <?php echo $emailErr;?></span>
```

```
<br><br>
Website:
<input type="text" name="website">
<span class="error"><?php echo $websiteErr;?></span>
<br><br>
<label>Comment: <textarea name="comment" rows="5" cols="40"></textarea>
<br><br>
Gender:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
<span class="error">* <?php echo $genderErr;?></span>
<br><br>
<input type="submit" name="submit" value="Submit">
 </form>
```

کد مورد نیاز برای این کد و اجباری کردن برخی بخش ها به صورت زیر است:

```
<!DOCTYPE HTML>
<html>
<head>
<style>
.error {color: #FF0000;}
</style>
</head>
<body>
 <?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";
 if ($_SERVER["REQUEST_METHOD"] == "POST")
{
   if (empty($_POST["name"]))
```

```
    {$nameErr = "Name is required";}
  else
    {$name = test_input($_POST["name"]);}
   if (empty($_POST["email"]))
    {$emailErr = "Email is required";}
  else
    {$email = test_input($_POST["email"]);}
   if (empty($_POST["website"]))
    {$website = "";}
  else
    {$website = test_input($_POST["website"]);}
   if (empty($_POST["comment"]))
    {$comment = "";}
  else
    {$comment = test_input($_POST["comment"]);}
   if (empty($_POST["gender"]))
    {$genderErr = "Gender is required";}
  else
    {$gender = test_input($_POST["gender"]);}
 }
 function test_input($data)
{
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data);
    return $data;
}
?>
 <h2>PHP Form Validation Example</h2>
<p><span class="error">* required field.</span></p>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
  Name: <input type="text" name="name">
```

```
  <span class="error">* <?php echo $nameErr;?></span>
  <br><br>
  E-mail: <input type="text" name="email">
  <span class="error">* <?php echo $emailErr;?></span>
  <br><br>
  Website: <input type="text" name="website">
  <span class="error"><?php echo $websiteErr;?></span>
  <br><br>
  Comment: <textarea name="comment" rows="5" cols="40"></textarea>
  <br><br>
  Gender:
  <input type="radio" name="gender" value="female">Female
  <input type="radio" name="gender" value="male">Male
  <span class="error">* <?php echo $genderErr;?></span>
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>
 <?php
echo "<h2>Your Input:</h2>";
echo $name;
echo "<br>";
echo $email;
echo "<br>";
echo $website;
echo "<br>";
echo $comment;
echo "<br>";
echo $gender;
?>
 </body>
</html>
```

معتبر بودن نام در PHP : کدی که در زیر نوشته ایم ساده ترین روش برای بررسی اعتبار نام نوشته شده از طرف کاربر است. این کد بررسی میکند که نام از حروف و فضای خالی استفاده شده باشد. در صورتی که بخش نام این ویژگی را دارا نبود خطا به کاربر نشان دهد. در این کد نویسی از تابع preg_match() نیز استفاده شده است. این تابع رشته های نوشته شده در الگو را بررسی میکند و اگر از الگوی مورد نظر پیروی میکرد آن را درست اعلام میکند و عدد ۱ را میفرستد، در غیر این صورت غلط بودن آن را برای نمایش خطا به بخش دستور بعدی با عدد ۰ میفرستند. این دستور درون علامت "/ /" و ویژگی های مورد بررسی در [ ] قرار میگیرند. اگر به دنبال بررسی یک رشته خاص باشد در انتهای تابع حرف i را قرار میدهند. اگر از حرف b قبل و بعد آن کلمه یا رشته استفاده شود تنها همان کلمه یا رشته را جستجو میکند و کلمات مشابه را در نظر نخواهد گرفت مانند: /bweb/b تنها web را جستجو میکند دیگر کلماتی مانند webmaster برایش اهمیت ندارد.

```
$name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z ]*$/",$name))
  {
  $nameErr = "Only letters and white space allowed";
  }
```

معتبر بودن E-mail در PHP : دستور زیر معتبر بودن آدرس ایمیل را بررسی میکند اگر درست نبود پیغام خطا را به کاربر نشان میدهد. نحوه نوشتن تابع preg_match() به صورت نوشتن یک ایمیل است. ابتدا بخش اول ایمیل و کاراکتر هایی که میتواند شامل این بخش شود را بیان کرده سپس با گذاشتن + و @ ویژگی کاراکترهایی که در این بخش از آدرس ایمیل قرار میگیرند و با + و . کاراکترهایی که ویژگی هایی مثل com، org و... را بیان کرده است. هرکدام از این حروف در این تابع بیان کننده یک ویژگی خاص است.

```
$email = test_input($_POST["email"]);
if (!preg_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$email))
  {
  $emailErr = "Invalid email format";
  }
```

معتبر بودن URL در PHP : دستور زیر آدرس سایتی را که در فرم مینوسید را بررسی میکند در صورت درست بودن و معتبر بودن URL آن را ذخیره نماید در غیر این صورت پیغام خطا به کاربر

نشان میدهد. در این بخش هم تابع(preg_match بخش های مختلف یک URL را بررسی میکند و در صورت درست بودن آن بخش به کاربر اجازه ارسال اطلاعات را میدهد.

```
$website = test_input($_POST["website"]);
if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
9+&@#\/%=~_|]/i",$website))
  {
  $websiteErr = "Invalid URL";
  }
```

حال میخواهیم اعتبارسنجی ایمیل و URL در PHP را در کنار هم مشاهده کنیم و آن ها را در کد نویسی فرم قبل اضافه نماییم. این بخش را در بین کد های else if قرار میدهیم.

```
<!DOCTYPE HTML>
<html>
<head>
<style>
.error {color: #FF0000;}
</style>
</head>
<body>
 <?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";
 if ($_SERVER["REQUEST_METHOD"] == "POST")
{
   if (empty($_POST["name"]))
     {$nameErr = "Name is required";}
   else
     {
     $name = test_input($_POST["name"]);
     // check if name only contains letters and whitespace
```

```
  if (!preg_match("/^[a-zA-Z ]*$/",$name))
    {
    $nameErr = "Only letters and white space allowed";
    }
  }
 if (empty($_POST["email"]))
  {$emailErr = "Email is required";}
 else
  {
  $email = test_input($_POST["email"]);
  // check if e-mail address syntax is valid
  if (!preg_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$email))
    {
    $emailErr = "Invalid email format";
    }
  }
 if (empty($_POST["website"]))
  {$website = "";}
 else
  {
  $website = test_input($_POST["website"]);
  // check if URL address syntax is valid (this regular expression also allows dashes in the URL)
  if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
9+&@#\/%=~_|]/i",$website))
    {
    $websiteErr = "Invalid URL";
    }
  }
 if (empty($_POST["comment"]))
  {$comment = "";}
 else
  {$comment = test_input($_POST["comment"]);}
```

```
  if (empty($_POST["gender"]))
    {$genderErr = "Gender is required";}
  else
    {$gender = test_input($_POST["gender"]);}
}
 function test_input($data)
{
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data);
    return $data;
}
?>
 <h2>PHP Form Validation Example</h2>
<p><span class="error">* required field.</span></p>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
  Name: <input type="text" name="name">
  <span class="error">* <?php echo $nameErr;?></span>
  <br><br>
  E-mail: <input type="text" name="email">
  <span class="error">* <?php echo $emailErr;?></span>
  <br><br>
  Website: <input type="text" name="website">
  <span class="error"><?php echo $websiteErr;?></span>
  <br><br>
  Comment: <textarea name="comment" rows="5" cols="40"></textarea>
  <br><br>
  Gender:
  <input type="radio" name="gender" value="female">Female
  <input type="radio" name="gender" value="male">Male
  <span class="error">* <?php echo $genderErr;?></span>
```

```
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>
 <?php
echo "<h2>Your Input:</h2>";
echo $name;
echo "<br>";
echo $email;
echo "<br>";
echo $website;
echo "<br>";
echo $comment;
echo "<br>";
echo $gender;
?>
 </body>
</html>
```

ساختن یک فرم برای آپلود فایل : فرم به سایت شما این امکان را می‌دهد که کاربران به خیلی ساده یک فایل را آپلود نمایند. در مثال زیر کد نویسی یک فرم HTML را برای آپلود فایل در PHP مشاهده می‌کنید.

```
<html>
<body>
 <form action="upload_file.php" method="post"
enctype="multipart/form-data">
<label for="file">Filename:</label>
<input type="file" name="file" id="file"><br>
<input type="submit" name="submit" value="Submit">
</form>
 </body>
</html>
```

نکات موجود در فرم HTML بالا:

- مشخصه enctype در برچسب فرم بیان کننده نوع محتوای فایلی است که می‌خواهد آپلود شود.
- multipart/form-data زمانی استفاده می‌شود که فایلی که می‌خواهد آپلود شود از جنس داده های باینری باشد.
- مشخصه "type="file در برچسب input زمانی استفاده می‌شود که ورودی به عنوان یک فایل پردازش شود. در حقیقت مسئول پردازش فرم است.

در نظر داشته باشید که بارگذاری فایل از طرف کاربران در سرور کاری پر خطر است. تنها به کاربرانی مطمئن اجزا آپلود یک فایل در PHP را بدهید.

ساختن اسکریپت هایی برای آپلود فایل در PHP

فایلی که در دستور HTML ساختیم به نام "upload_file.php" است. این فایل شامل کد هایی برای آپلود فایل در PHP است. این کد ها به صورت زیر است:

```
<?php
if ($_FILES["file"]["error"] > 0)
  {
  echo "Error: " . $_FILES["file"]["error"] . "<br>";
  }
else
  {
  echo "Upload: " . $_FILES["file"]["name"] . "<br>";
  echo "Type: " . $_FILES["file"]["type"] . "<br>";
  echo "Size: " . ($_FILES["file"]["size"] / 1024) . " kB<br>";
  echo "Stored in: " . $_FILES["file"]["tmp_name"];
  }
?>
```

در مثال بالا کد PHP به صورت شرطی نوشته است که اگر متغیر گلوبال $_file دارای خطا بود، نمایش دهد که خطا وجود دارد و اطلاعات دیگر را نمایش ندهد در غیر این صورت یعنی اگر متغیر هیچ خطایی را شامل در بر نداشت، نام فایل، نوع فایل، سایز وکپی نام فایلی که به صورت

موقتی در سرور ذخیره شده است را نمایش می‌دهد. پارامترهایی که در ورودی فایل استفاده می‌شود به صورت زیر است:

$_FILES["file"]["name"] :نام فایلی که آپلود شده است

$_FILES["file"]["type"]:نوع فایلی که آپلود شده است

$_FILES["file"]["size"]: سایز فایلی که آپلود شده است

$_FILES["file"]["tmp_name"]:کپی نام فایلی که به صورت موقتی در سرور ذخیره شده است

محدودیت آپلود فایل در PHP : در این اسکریپت ها یک سری محدودیت ها برای آپلود فایل ها قرار می‌دهیم. در این مثال تنها فایلی هایی با فرمت های .jpeg .gif و . png و حداکثر تا ۲۰ کیلو بایت می‌توان آپلود کرد.

```
<?php
$allowedExts = array("gif", "jpeg", "jpg", "png");
$temp = explode(".", $_FILES["file"]["name"]);
$extension = end($temp);
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/jpg")
|| ($_FILES["file"]["type"] == "image/pjpeg")
|| ($_FILES["file"]["type"] == "image/x-png")
|| ($_FILES["file"]["type"] == "image/png"))
&& ($_FILES["file"]["size"] < 20000)
&& in_array($extension, $allowedExts))
  {
  if ($_FILES["file"]["error"] > 0)
    {
    echo "Error: " . $_FILES["file"]["error"] . "<br>";
    }
  else
    {
```

```
    echo "Upload: " . $_FILES["file"]["name"] . "<br>";
    echo "Type: " . $_FILES["file"]["type"] . "<br>";
    echo "Size: " . ($_FILES["file"]["size"] / 1024) . " kB<br>";
    echo "Stored in: " . $_FILES["file"]["tmp_name"];
    }
  }
else
  {
  echo "Invalid file";
  }
?>
```

در این دستور یک آرایه تعریف شده است که اگر فایلی که می‌خواهیم بارگذاری شود باید یکی از این فرمت ها را دارا باشد و سایز این فایل کمتر از ۲۰۰۰۰ هزار بایت باشد و در آخر if اول نوشته شده است" و اگر پسوند فایل در آرایه جزو پسوندهای معرفی شده باشد شرط زیر را بررسی کن"

extension یک متغییری به نام Temp تعریف کرده است و آخرین temp مورد نظرش است(فرمت فایل). متغیر temp می‌گوید اگر نام فایل دارای چندین آرایه بود آن را با علامت" . " از هم جدا کن. در حقیقت آخرین آرایه بعد از نقطه همان فرمت فایل می‌شود.

ذخیره کردن فایل آپلود شده : هنگامی که اسکریپت ها به پایان رسیدند یک کپی به صورت موقت از فایل نمایان می‌شود. برای این که فایل آپلود شده را ذخیره کنیم نیاز داریم آن را در یک جای دیگر کپی کنیم

```
<?php
$allowedExts = array("gif", "jpeg", "jpg", "png");
$temp = explode(".", $_FILES["file"]["name"]);
$extension = end($temp);
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/jpg")
|| ($_FILES["file"]["type"] == "image/pjpeg")
```

```
|| ($_FILES["file"]["type"] == "image/x-png")
|| ($_FILES["file"]["type"] == "image/png"))
&& ($_FILES["file"]["size"] < 20000)
&& in_array($extension, $allowedExts))
  {
  if ($_FILES["file"]["error"] > 0)
    {
    echo "Return Code: " . $_FILES["file"]["error"] . "<br>";
    }
  else
    {
    echo "Upload: " . $_FILES["file"]["name"] . "<br>";
    echo "Type: " . $_FILES["file"]["type"] . "<br>";
    echo "Size: " . ($_FILES["file"]["size"] / 1024) . " kB<br>";
    echo "Temp file: " . $_FILES["file"]["tmp_name"] . "<br>";
     if (file_exists("upload/" . $_FILES["file"]["name"]))
      {
      echo $_FILES["file"]["name"] . " already exists. ";
      }
    else
      {
      move_uploaded_file($_FILES["file"]["tmp_name"],
      "upload/" . $_FILES["file"]["name"]);
      echo "Stored in: " . "upload/" . $_FILES["file"]["name"];
      }
    }
  }
else
  {
  echo "Invalid file";
  }
?>
```

مثال قبل همان مثال بالاست با این تفاوت که می‌خواهیم آن را ذخیره کنیم. اگر فایل وجود داشت نمایش بده که فایل وجود دارد و اگر وجود نداشت یک کپی از آن را ذخیره کن و نام آن را آپلود بساز.

کوکی در PHP

کوکی در PHP یک فایل بسیار کوچک است که سرور بر روی مرورگر کامپیوتر کاربر ایجاد می‌کند و اطلاعات کاربر را در هر بار ورود به آن سایت در اختیار سرور می‌گذارد. این اطلاعات می‌تواند شامل نام کاربری، رمز عبور و تنظیماتی که کاربر بر روی سایت برای خود ایجاد کرده‌اند باشد. برای ذخیره این اطلاعات معمولا از کاربر در این باره سوال پرسیده می‌شود که می‌خواهد اطلاعاتش را به خاطر سپرده شود یا خیر.

ایجاد کردن یک کوکی در PHP بسیار ساده است، تنها کافی است از تابع setcookie() استفاده کنید. اما باید این نکته مورد توجه قرار دهید که این تابع باید قبل از برچسب <html> قرار بگیرد.

نحوه نگارش کوکی:

```
setcookie(name, value, expire, path, domain);
```

در مثال زیر یک کوکی به نام " user" ایجاد می‌کنیم و "Alex Porter" را به عنوان ارزش برایش در نظر می‌گیریم. همچنین برایش این ویژگی را در نظر می‌گیریم که این کوکی بعد از ۱ ساعت دیگر قابل استفاده نباشد.

```
<?php
setcookie("user", "Alex Porter", time()+3600);
?>

<html>
.....
```

ارزش کوکی به صورت خودکار در URL ذخیره می‌شود و به صورت خودکار از بین می‌رود. اطلاعات کوکی را می‌توان به نحو دیگری نیز غیر قابل استفاده کرد. در مثال زیر این روش را توضیح می‌دهیم. لازم به ذکر است که این روش ساده تر است و می‌توان ثانیه ها را نیز لحاظ کرد.

```
<?php
$expire=time()+60*60*24*30;
setcookie("user", "Alex Porter", $expire);
?>

<html>
.....
```

زمان استفاده از این کوکی تنها برای یک ماه است، این زمان را به صورت ۶۰ ثانیه، ۶۰ دقیقه، ۲۴ ساعت، ۳۰ روز نمایش داده شده است.

بازیابی ارزش یک کوکی : برای بازیابی ارزش یک کوکی کافی است از متغیر $_cookie استفاده کنید. در مثال زیر از تابع isset() برای بررسی کوکی استفاده می‌شود. این که آیا برای این کاربر کوکی اعمال شده است یا خیر.

```
<html>
<body>
 <?php
if (isset($_COOKIE["user"]))
  echo "Welcome " . $_COOKIE["user"] . "!<br>";
else
  echo "Welcome guest!<br>";
?>
</body>
</html>
```

اگر در بالا کاربر قبلا عضو شده باشد با نام او به خوش آمد می‌گوید در غیر این صورت به عنوان مهمان سایت به او خوش آمد می‌گوید.

پاک کردن یک کوکی : هنگامی که می‌خواهید یک کوکی را پاک نمایید باید اطمینان حاصل نمایید تا کوکی مورد نظر تاریخ استفاده اش به پایان رسیده باشد.

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time()-3600);
?>
```

در این مثال کوکی تنها ۱ ساعت قابلیت استفاده دارد. اگر دقت کنید قبل از بیان زمان به جای علامت + از علامت – استفاده شده است و در این زمان کوکی پاک خواهد شد.

پشتیبانی مرورگر از کوکی : اگر برنامه شما جزو برنامه هایی باشد که مرورگر کوکی را پشتیبانی نمی‌کند مجبور هستید از روش دیگری برای ذخیره کوکی استفاده کنید

فرم زیر کاربر را در صورت کلیک بر روی دکمه submit به فایل "welcome.php" می‌فرستد.

```
<html>
<body>
 <form action="welcome.php" method="post">
Name: <input type="text" name="name">
Age: <input type="text" name="age">
<input type="submit">
</form>
 </body>
</html>
```

و برای بازیابی ارزش فایل "welcome.php" به صورت زیر عمل می‌کنیم:

```
<html>
<body>
 Welcome <?php echo $_POST["name"]; ?>.<br>
You are <?php echo $_POST["age"]; ?> years old.
 </body>
```

```
</html>
```

در این دستور ارزش نوشته شده در فایل "welcome.php" ذخیره می‌شود و برای بازیابی باید به این فایل رجوع کرد. و دیگر کوکی تعریف نمی‌شود.

جلسه یا Sessions

: در PHP برای ذخیره اطلاعات کاربر استفاده می‌شود. در کل فرایند سشن در کامپیوتر به معنای باز کردن یک نرم افزار و ایجاد یک سری تغییرات در نرم افزار و بستن آن است. اما در سیستم وب و اینترنت این موضوع به صورت دیگری رخ می‌دهد.

هاست یا سرور سایتی که شما بازدید کننده آن هستید، نمی‌تواند اطلاعاتی راجع به شما به سرور بدهد زیرا مرورگر HTML هیچ اطلاعاتی راجع به شما نمی‌تواند بگوید. و در این شرایط تمایز کاربران از هم دچار مشکل می‌شود و سرور نمی‌تواند به شما خدماتی ارائه دهد. برای حل این مشکل، برنامه نویسی PHP بخشی به نام Sessions را تعریف نموده است. اطلاعات شما با استفاده از این دستور در سرور سایتی که از آن بازدید می‌کنید ذخیره می‌شود. بنابراین برای مشاهده صفحات مختلف یک سایت نیاز نیست در هر صفحه نام کاربری و رمز عبور خود را وارد نمایید. راحتی ای که شما در بازدید صفحات یک سایت دارید به دلیل این دستور است.

هاست سایتی که به ان وارد می‌شوید در لحظه ورود یک Sessions خاص به نام unique ID – UID برایتان می‌سازد و این اطلاعات را در سشن ذخیره می‌کند. این اطلاعات تا زمانی که در سایت هستید یا Sessions دارای اعتبار است در سرور ذخیره می‌شود همچنین این اطلاعات در کوکی نیز ذخیره می‌شود.

Sessions نیز همانند کوکی ها موقتی هستند و تا یک زمان خاص ذخیره می‌شود.

شروع Sessions در PHP :

قبل از ذخیره شدن اطلاعات در سشن، شما باید Sessions را شروع کنید. برای این کار باید دقت داشته باشید که تابع session_start() باید قبل از برچسب <html> نوشته شود.	<?php session_start(); ?> <html> <body> </body> </html>

این کد به شما اجازه می‌دهد زمانی که کاربر در سایت ثبت نام می‌کند،اطلاعاتش ذخیره شود و یک UID برای session کاربر می‌سازد.

ذخیره متغیرهای session : درست ترین راه برای ذخیره و بازیابی اطلاعات درون سشن استفاده از متغیر $_SESSION در PHP است.

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>
 <html>
<body>
 <?php
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>
 </body>
</html>
```

این دستور می‌خواهد تعداد دفعات نمایش صفحه را مشخص کند بنابراین خروجی به صورت زیر است: Pageviews=1

مثال زیر یک دستور است که با تعداد دفعات بازدید یک صفحه را نشان می‌دهد. با استفاده از تابع isset() متغیری به نام view را که در session ذخیره شده است را بررسی می‌کند، اگر کاربر برای بار اول وارد سایت شده است عددی برابر با ۱ را نمایش می‌دهد و در ورود های بعدی مقدار آن را افزایش می‌دهد.

```
<?php
session_start();
 if(isset($_SESSION['views']))
$_SESSION['views']=$_SESSION['views']+1;
else
$_SESSION['views']=1;
echo "Views=". $_SESSION['views'];
?>
```

پاک کردن یک session : هنگامی که بخواهید اطلاعاتی را از یک session پاک کنید، می‌توانید از تابع های unset() و session_destroy() استفاده کنیم. تابع unset() برای آزاد کردن یک متغیر در session استفاده می‌شود.

```
<?php
session_start();
if(isset($_SESSION['views']))
  unset($_SESSION['views']);
?>
```

برای پاک کردن کامل یک session از تابع session_destroy() استفاده می‌شود:

<?php session_destroy(); ?>	باید دقت داشته باشید این دستور تمام اطلاعات درون session را پاک می‌کند.

معرفی تابع E-mail و چگونگی ارسال ایمیل در PHP

از فرم ها در سایت برای عضویت کاربران، ایجاد خبرنامه ، تماس با ما و ... استفاده میشود و همه این فرم ها دارای یک بخش ایمیل هستند. با استفاده از این بخش میتوان ایمیلی مبتنی بر عضویت در سایت یا خبر برای کاربر فرستاد. تابع Email() این امکان را در PHP ایجاد میکند.

این تابع به صورت روبرو نوشته میشود: mail(to,subject,message,headers,parameters)

. این وظایف

به شرح زیر است:

- to : نوشتن این پارامتر اجباری است و ایمیل گیرنده و یا گیرندگان را مشخص میکند.
- subject : نوشتن این پارامتر اجباری است و از نوع متن بوده و تنها میتواند یک خط باشد. موضوع عنوان ایمیل را مشخص میکند

- message : نوشتن این پارامتر اجباری است و از نوع متن بوده و نباید بیش از ۷۰ کاراکتر باشد. متن نوشته شده داخل ایمیل را مشخص میکند و هر خط باید با کاراکتر (/n) از خط قبل و بعد جدا شود.

- header : نوشتن این پارمتر اختیاری است. Cc یا Bcc را مشخص میکند. برای جدا کردن هر خط از خط قبلی باید از کارکتر (r\n\) استفاده شود.

- Parametrs : نوشتن این بخش اختیاری است. برای اضافه کردن پارامتر ها به برنامه ارسال ایمیل استفاده میشود.

ارسال ایمیل : کد نویسی زیر ساده ترین روش برای ارسال یک ایمیل از طریق PHP است. البته فرستادن یک ایمیل به این سادگی بعید است در ادامه به روش کامل فرستادن ایمیل از طریق PHP را آشنا خواهید شد.

در مثال زیر متغیر های to، $subject، $message، $header، $form تعریف شده اند و از این متغیر ها در تابع Email() استفاده شده است. این تابع ایمیل را ارسال کرده و یک پیام مبتنی بر تایید برای کاربر میفرستد.

```
<?php
$to = "someone@example.com";
$subject = "Test mail";
$message = "Hello! This is a simple email message.";
$from = "someonelse@example.com";
$headers = "From:" . $from;
mail($to,$subject,$message,$headers);
echo "Mail Sent.";
?>
```

بخش ایمیل در یک فرم PHP : مثال زیر بخشی از یک فرم که شامل ایمیل است را نشان میدهد. در این مثال متغیر ها اطلاعات را از کاربر گرفته و در متغیر $_REQUEST ذخیره میکند. در If بررسی میکند که پارامترهای دلخواه در این متغیر برای ذخیره شدن نوشته شده اند یا خیر اگر

نوشته شده بودند که ذخیره کرده و ایمیلی برای کاربر میفرستند در غیر این صورت دوباره آن فرم را نمایش میدهد.

```
<html>
<body>
 <?php
if (isset($_REQUEST['email']))
//if "email" is filled out, send email
  {
  //send email
  $email = $_REQUEST['email'] ;
  $subject = $_REQUEST['subject'] ;
  $message = $_REQUEST['message'] ;
  mail("someone@example.com", $subject,
  $message, "From:" . $email);
  echo "Thank you for using our mail form";
  }
else
//if "email" is not filled out, display the form
  {
  echo "<form method='post' action='mailform.php'>
  Email: <input name='email' type='text'><br>
  Subject: <input name='subject' type='text'><br>
  Message:<br>
  <textarea name='message' rows='15' cols='40'>
  </textarea><br>
  <input type='submit'>
  </form>";
  }
?>
 </body>
</html>
```

این روشی است برای فرستادن ایمیل به کاربر، اما لازم به ذکر است که این کد از امنیت برخودار نیست. می خواهیم بررسی کنیم که ایمیل نوشته شده اسپم هست یا یک کاربر واقعی است.

```
<html>
<body>
 <?php
if (isset($_REQUEST['email']))
//if "email" is filled out, send email
  {
  //send email
  $email = $_REQUEST['email'] ;
  $subject = $_REQUEST['subject'] ;
  $message = $_REQUEST['message'] ;
  mail("someone@example.com", "Subject: $subject",
  $message, "From: $email" );
  echo "Thank you for using our mail form";
  }
else
//if "email" is not filled out, display the form
  {
  echo "<form method='post' action='mailform.php'>
  Email: <input name='email' type='text'><br>
  Subject: <input name='subject' type='text'><br>
  Message:<br>
  <textarea name='message' rows='15' cols='40'>
  </textarea><br>
  <input type='submit'>
  </form>";
  }
?>
 </body>
</html>
```

مشکل کد بالا این است که کاربر میتواند در بخش Header تابع ایمیل داده های غیر مجاز وارد نماید. تصور کنید که کاربر تعداد زیادی ایمیل مانند زیر را در این بخش وارد نماید چه اتفاقی پیش می آید؟

```
someone@example.com%0ACc:person2@example.com
%0ABcc:person3@example.com,person3@example.com,

anotherperson4@example.com,person5@example.com
%0ABTo:person6@example.com
```

هر آنچه که تابع ایمیل میخواهد برای کابر بفرستند در header ذخیره شده و برای تمام ایمیل های فرستاده میشود و از نظر وب مستر دارای ایراد است و امنیت ایمیل در PHP و باعث ایجاد اختلال در برنامه میشود.

جلوگیری از حملات به ایمیل در PHP :

برای جلوگیری این حملات اسکریپتی به ایمیل در PHP به معتبر بودن داده ها را بررسی میکنیم. برای افزایش امنیت ایمیل در PHP به کد قبل دو تابع اضافه کردیم. یک تابع بررسی میکند و تعداد ایمیل ها اگر زیاد باشد آن ها را پاک میکند دیگری معتبر بودن ان را بررسی میکند. در صورتی که این شرایط برقرار نباشد به کاربر پیغام ایمیل معتبر را وارد نمایید را نشان میدهد در غیر این صورت اطلاعات ایمیل را ذخیره کرده و ایمیلی با پیغام نوشته شده در message به کاربر ارسال میشود.

- تابع ( ) FILTER_SANITIZE_EMAIL: اگر بیش از یک ایمیل نوشته شده باشد، ایمیل ها را پاک میکند.
- تابع ( ) FILTER_VALIDATE_EMAIL: معتبر بودن ایمیل ها را بررسی میکند.

```
<html>
<body>
<?php
function spamcheck($field)
  {
  //filter_var() sanitizes the e-mail
```

```
  //address using FILTER_SANITIZE_EMAIL
  $field=filter_var($field, FILTER_SANITIZE_EMAIL);

  //filter_var() validates the e-mail
  //address using FILTER_VALIDATE_EMAIL
  if(filter_var($field, FILTER_VALIDATE_EMAIL))
    {
    return TRUE;
    }
  else
    {
    return FALSE;
    }
  }

if (isset($_REQUEST['email']))
  {//if "email" is filled out, proceed

  //check if the email address is invalid
  $mailcheck = spamcheck($_REQUEST['email']);
  if ($mailcheck==FALSE)
    {
    echo "Invalid input";
    }
  else
    {//send email
    $email = $_REQUEST['email'] ;
    $subject = $_REQUEST['subject'] ;
    $message = $_REQUEST['message'] ;
    mail("someone@example.com", "Subject: $subject",
    $message, "From: $email" );
    echo "Thank you for using our mail form";
```

```
    }
  }
else
  {//if "email" is not filled out, display the form
  echo "<form method='post' action='mailform.php'>
  Email: <input name='email' type='text'><br>
  Subject: <input name='subject' type='text'><br>
  Message:<br>
  <textarea name='message' rows='15' cols='40'>
  </textarea><br>
  <input type='submit'>
  </form>";
  }
?>

</body>
</html>
```

به طور پیش فرض مدیریت خطا بسیار ساده است. پیام خطا دارای یک نام فایل، شماره ی خطی که در آن خطا رخ داده است و توضیحی درباره خطا به مرورگر برای کاربر فرستاده میشود.

هنگامی که اسکریپت یا یک برنامه جدید به وب اضافه میکنید، مدیریت خطا از مهمترین بخش ها به شمار میرود. اگر در مدیریت خطا دارای نقص باشید برنامه و اسکریپت هایی که نوشته اید بسیار ابتدایی به نظر آمده و حتی میتواند شبکه را در معرض خطر قرار دهد.

روش هایی برای مدیریت خطا در PHP وجود دارد که روش های رایج آن به صورت زیر است:

- تابع ساده die()
- خطا های معمول و فعال کننده های خطا
- گزارش خطا

مدیریت خطا در PHP : ابتدایی ترین و ساده ترین راه برای مدیریت خطا در PHP استفاده از تابع die() است. در مثال زیر میخواهیم یک فایل متنی را باز کنیم. در صورت نبودن فایل پیغام خطا به کاربر نشان میدهد.

```
<?php
$file=fopen("welcome.txt","r");
?>
```

با استفاده از تابع fopen فایل مورد نظر را باز کردیم، اگر تابع دارای خطا باشد خطا را به صورت زیر نشان میدهد:

```
Warning: fopen(welcome.txt) [function.fopen]: failed to open stream:
No such file or directory in C:\webfolder\test.php on line 2
```

برای این که خطایی مطابق با خطای فوق را نداشته باشیم و مطمئن باشیم که ابتدا بررسی شده است که فایلی به این نام وجود داشته است یا خیر و در صورت پیدا نکردن فایل آن را نشان دهد کد زیر میتواند به ما کمک کند:

```
<?php
if(!file_exists("welcome.txt"))
  {
  die("File not found");
  }
else
  {
  $file=fopen("welcome.txt","r");
  }
?>
```

در ابتدای این کد گفته شده اگر فایل welcome.txt وجود نداشت تابع die() پیام خطا را نشان میدهد، در غیر این صورت فایل را باز میکند. پیام خطایی که نشان میدهد دیگر همانند پیام خطای قبل طولانی نخواهد بود و تنها بیان میکند که فایلی یافت نشد. File not found

نه تر باید از

دستورهای کامل تری استفاده شود. ایجاد یک کنترل‌کننده خطا در PHP بسیار ساده است. می توان یک تابع ساده ایجاد کرد تا زمانی که خطایی ایجاد شد، فراخوانی شود.

این تابع باید حداقل دو پارامتر داشته باشد. پارامترهای اجباری :

- error level(اهمیت خطا)
- error message(پیغام خطا)

اما در کل میتواند ۵ پارامتر را در برگیرد. نوشتن سه پارامتر دیگر به صورت اختیاری است. این پارمترها شامل:

- line-number (شماره خطی که در آن خطا رخ داده است)
- the error context (محتوانی خطا)
- file (نام فایل)

نحوه نوشتن این تابع بری ایجاد کنترل‌کننده خطا در PHP به صورت زیر است:

```
error_function(error_level,error_message,error_file,error_line,error_context)
```

پارامترها	توضیحات
error_level	این پارامتر اهمیت خطا را نشان میدهد.نوشتن اجباری است و با عدد نمایش میدهند. توضیح آن در جدول زیر آمده است.
error_message	پیغامی درباره خطا است. نوشتن آن اجباری است
error_file	این پیغام نام متغیر را مشخص میکند و نوشتن آن اختیاری است.
error_line	این پیغام شماره خطی را که خطا در آن رخ داده است نشان میدهد. نوشتن این پارامتر اختیاری است
error_context	این خطا شامل تمام متغیرها و ارزش آن ها است زمانی که خطایی رخ میددهد. نوشتن این پارامتر اختیاری است.

گزارش اهمیت خطا درPHP :

جدول زیر انواع اهمیت خطا در PHP وتوضیح هر یک را نشان میدهد.

توضیحات	پارامتر	ارزش
اهمیت خطا کم بوده و مانع اجرای کد نویسی ها نمیشود.	E_WARNING	۲
برنامه ممکن اسن دچار مشکل شده باشد و خطا نمایش دهد اما در روند ادامه برنامه مشکلی ایجاد نمیکند.	E_NOTICE	۸
یک پیغام مهم است و در روند اجرای اسکریپت ها اختلالی ایجاد میکند.	E_USER_ERROR	۲۵۶
یک پیغام ساده است و در روند اجرای اسکریپت ها اختلالی ایجاد نمیکند. مانند E_WARNING خطا هم توسط طراح وب برنامه پیش بینی میشود	E_USER_WARNING	۵۱۲
پیغام خطا یک پیغام معمولی بوده و طراح وب آن را پیش بینی میکند	E_USER_NOTICE	۱۰۲۴
یک خطای بسیار مهم و اسکریپت ها را از کار می اندازد	E_RECOVERABLE_ERROR	۴۰۹۶
همه نوع خطا را شامل میشود.	E_ALL	۸۱۹۱

حال بیاید برای کنترل خطا ها یک تابع را به عنوان مثال تعریف کنیم:

```
function customError($errno, $errstr)
  {
  echo "<b>Error:</b> [$errno] $errstr<br>";
  echo "Ending Script";
  die();
  }
```

کد نوشته شده فوق به صورت ساده ای کنترل کننده خطا در PHP است. در این تابع متغیر اول $errno اهمیت خطا و $errstr پیغام خطا را ذخیره میکند و سپس به خروجی میفرستد.

کنترل کننده خطا در PHP به طور پیش فرض وجود ایجاد شده است، حال میخواهیم تابعی را ایجاد کنیم که به طور پیش فرض خطا ها را در طول اجرای اسکریپت ها کنترل کند. همچنین این امکان وجود دارد که برای کدها و اسکریپت های مختلف به صورت پیش فرض کنترل کننده خطا تعریف کنید به همین دلیل میتوانید به شخصی سازی کنترل کننده خطا بپردازید.

شخصی سازی کنترل کننده در PHP: در مثال زیر از با شخصی سازی کنترل کننده خطا برای تمام خطا ها استفاده میکنیم:

```
set_error_handler("customError");
```

تا زمانی که از تابع set-error-handel برای شخصی سازی تمام خطا ها استفاده میشود فقط نیاز به یک پارامتر است، البته خودتان میتوانید یک پارامتر دیگر اضافه کنید تا اهمیت خطا را برایتان مشخص کند. به مثال زیر دقت کنید:

```
<?php
//error handler function
function customError($errno, $errstr)
  {
  echo "<b>Error:</b> [$errno] $errstr";
  }
 //set error handler
set_error_handler("customError");
 //trigger error
echo($test);
?>
```

خطای این مثال تعریف نشدن متغیر test است. در این مثال هیچ تعریفی برای متغیر test نشده است. به همین دلیل خطا خواهد گرفت. خروجی این دستور باید چیزی شبیه به این باشد:

```
Error: [8] Undefined variable: test
```

تابع trigger_error() : این تابع در جاهایی که کاربر میخواهد داده ای وارد کند بسیار مفید است. این تابع از خطاهای احتمالی را پیش‌بینی میکند و از ایجاد اشکال در روند برنامه جلوگیری میکند.

در مثال این دستور را در کنار دستور هایی که در مبحث های ایجاد کنترل‌کننده خطا و مدیریت خطا در PHP مینویسیم تا اهمیت خطا و ویژگی های آن را نشان دهد. این تابع میتواند اهمیت خطاهای E_USER_NOTICE،E_USER_WARNING،E_USER_ERROR و ارزش این خطا ها را که در درس ایجاد کنترل کننده خطا توضیح دادیم را در پیغام خطا بیان کند.

```
<?php
//error handler function
function customError($errno, $errstr)
  {
  echo "<b>Error:</b> [$errno] $errstr<br>";
  echo "Ending Script";
  die();
  }
 //set error handler
set_error_handler("customError",E_USER_WARNING);
 //trigger error
$test=2;
if ($test>1)
  {
  trigger_error("Value must be 1 or below",E_USER_WARNING);
  }
?>
```

تابع trigger_error() در PHP این خطا را مدیریت کرده و پیغام خطا میدهد. خروجی به صورت زیر است:

```
Error: [512] Value must be 1 or below
Ending Script
```

تابع Error-Log() : به وسیله تابع Error-Log()، می توانید پیام خطاهایی که اتفاق افتاده اند را به یک فایل خاص یا به ایمیل کاربر ارسال نمایید . ارسال یک ایمیل حاوی اطلاعات خطا می تواند یکی از کارآمدترین روش ها باشد. به یاد داشته باشید که این کار را برای خطا های معمولی انجام ندهید. به صورت پیش فرض ، PHP یک پیام حاوی اطلاعات خطاهای اتفاق افتاده را به سیستم ورود به سرور یا یک فایل خاص، بسته به اینکه فایل php.ini ، بر روی سرور چگونه تنظیم شده باشد، میفرستد.

```
<?php
//error handler function
```

```
function customError($errno, $errstr)
  {
  echo "<b>Error:</b> [$errno] $errstr<br>";
  echo "Webmaster has been notified";
  error_log("Error: [$errno] $errstr",1,
  "someone@example.com","From: webmaster@example.com");
  }
 //set error handler
set_error_handler("customError",E_USER_WARNING);
 //trigger error
$test=2;
if ($test>1)
  {
  trigger_error("Value must be 1 or below",E_USER_WARNING);
  }
?>
```

در این مثال در صورتی که خطای خاصی رخ داده باشد ایمیلی شامل پیغام خطا برای کاربر ارسال میشود. خطا در این مثال متغیر test از میزان مقدار مجاز بیشتر مقدار گرفته است.

تابع رشته ای chop() : این تابع فضاهای خالی و کاراکترهای مشخص شده را از سمت راست کارکترهای رشته ای حذف میکند.این تابع برای PHP4 به بالا قابل استفاده است. این تابع به صورت زیر نوشته میشود:

```
chop(string,charlist)
```

نوشتن بخش string مهم و ضروری است اما مشخص کردن کارکتر مورد نظر به صورت دلخواه است. اگر کاراکتر خاصی را انتخاب نکنید تنها فضاهای خالی حذف خواهد شد. در جدول زیر با لیست کاراکترهایی که به صورت خودکار توسط این تابع حذف میشود، آشنا خواهید شد.

کاراکتر	توضیحات
NULL – ۰"\"	تنها کاراکتر آن شمارش میشود اما ارزشی برایش قرار نمی دهد و نشان داده نمی شود

t″ – tab\″	به اندازه ۴ کاراکتر فضای خالی ایجاد میکند
n″ – new line\″	خط جدید اضافه میکند
x0B″ – vertical tab\″	به صورت عمودی ۴ کاراکتر اضافه میکند
ordinary white space – ″ ″	یک فضای خالی ایجاد میکند

<?php $str = "Hello World!\n\n"; echo $str; echo chop($str); ?>	خروجی بدون chop() Hello World! Hello World!	خروجی با chop() Hello World! Hello World!

در این مثال کارکتر n/ باعث شده است که این دو جمله با فاصله دو خط از هم چاپ شوند.

<!DOCTYPE html> <html> <body> <?php $str = "Hello World!"; echo $str . " "; echo chop($str,"World!"); ?> </body> </html>	خروجی Hello World! Hello همان طور که مشاهده میکنید با استفاده از تابع رشته ای chop() کاراکتر world! از جمله دوم حذف شده است.

تابع html_entity_decode() : کد های HTML ای که بخواهید را به کاراکتر تبدیل کرده و باعث میشود کاراکترهای مورد نظرتان را به راحتی مشاهده کنید.

```
<?php
$str = "&lt;&copy; W3S&ccedil;h&deg;&deg;&brvbar;&sect;&gt;";
echo html_entity_decode($str);
?>
```

همان طور که مشاهده میکنید متغیر $str کد های HTML را شامل میشود، سپس خواسته شده با استفاده از تابع html_entity_decode() متغیر $str چاپ شود. خروجی به صورت زیر خواهد شد: w3sh

هر آنچه که حذف شد جزو کد های HTML بوده است.

```
<?php
$str = "Jane &amp; &#039;Tarzan&#039;";
echo html_entity_decode($str, ENT_COMPAT); // Will only convert double quotes
echo "<br>";
echo html_entity_decode($str, ENT_QUOTES); // Converts double and single quotes
echo "<br>";
echo html_entity_decode($str, ENT_NOQUOTES); // Does not convert any quotes
?>
```

قبل از تحلیل دستور نوشته شده نحوه نوشتن تابع تابع html_entity_decode() را بررسی کنیم. درون این تابع بخش های زیر قرار میگیرد. html_entity_decode(string,flags,character-set)

بخش string اجباری است. حتما باید مشخص شود که چه متغیری باید این تغییرات بر رویش اعمال شود.

بخش flag و character-set اختیاری هستند. که در این مثال از بخش flag برای محدود کردن تابع استفاده شده است. حال echo ها نوشته شده در این مثال را قدم به قدم بررسی میکنیم:

```
echo html_entity_decode($str, ENT_COMPAT);
```

متغیر str مشخص است. ENT_COMPAT یعنی دبل کتیشون ها را تبدیل کن. دبل کتیشون در HTML به صورت " نوشته میشود. در این مثال چنین کدی وجود ندارد بنابراین تغییری در چاپ متغیر ایجاد نمیشود : ENT_QUOTES

کدهای HTML مربوط به تک کتیشن را به کاراکتر تبدیل میکند. ' یا تک کتیشن در کد های HTML به صورت زیر نمایش داده میشوند. 039# پس در این echo این علامت برداشته میشود. و به صورت زیر چاپ میشود: Jane & 'Tarzan'

و echo اخر که از ENT_NOQUOTES استفاده شده است. این flag هیچ یک از کد های HTML را تبدیل نمی کند. و خروجی به صورت زیر خواهد شد: Jane & 'Tarzan'

در کل همه این چاپ ها در HTML به این صورت خواهد بود:

<!DOCTYPE html> <html> <body> Jane & 'Tarzan' Jane & 'Tarzan' Jane & 'Tarzan' </body> </html>	خروجی نهایی کد HTML Jane & 'Tarzan' Jane & 'Tarzan' Jane & 'Tarzan'

تابع implode() : تابع implode() دارای دو بخش است. بخش separator اختیاری است هر آنچه که میخواهید بین رشته های قرار بگیرد را میتوانید اینجا تعریف کنید. این بخش میتواند با گذاشتن " " خالی باشد یا " +" علامت قرار دهید. بخش array اجباری است. باید آن آرایه ای که میخواهید به رشته تغییر کند را تعریف کنید. implode(separator,array)

<!DOCTYPE html> <html> <body> <?php $arr = array('Hello','World!','Beautiful','Day!'); echo implode(" ",$arr); ?> </body> </html>	دراین مثال آرایه ای تعریف شده است و میخواهیم بخش های این آرایه را به صورت رشته تعریف کنیم. با استفاده از تابع implode() این عمل صورت می پذیرد.در این مثال میخواهیم بین رشته ها فضای خالی باشد بنابراین بخش اول را با " " خالی میگذاریم. خروجی به صورت زیر خواهد بود: Hello World! Beautiful Day!

پایگاه داده : پایگاه داده برای ذخیره اطلاعات بسیار مناسب است. یک شرکت معمولا دارای پایگاه های داده زیر هستند:

۱ - کارکنان ۲ - مشتریان ۳- محصولات ۴ - سفارش ها

با استفاده از PHP میتوانید به پایگاه داده دسترسی داشته باشید و اطلاعاتی که میخواهید را تغییر دهید. مای اس کیو ال یکی از معروف ترین سیستم های پایگاه داده است که با پی اچ پی کار میکند.

MYSQL چیست : MYSQL یک سیستم پایگاه داده که در وب قابل استفاده بوده و قابل اجرا بر روی سرور است و سریع، قابل اعتماد و استاندارد های SQL پشتیبانی میکند ، بر روی تعداد زیادی از سیستم عامل ها کار میکند و توسط شرکت اوراکل توزیع شده و پشتیبانی میشود. داده ها در MYSQL در یک جدول ذخیره میشود. جدول شامل داده های مرتبط به هم است که در سطر و ستون ها به صورت منظم قرار گرفته اند.

ایجاد ارتباط با سرور MYSQL : قبل از دسترسی به پایگاه داده باید ارتباط با سرورMYSQL ایجاد کرد. برای ایجاد ارتباط با سرور MYSQL از کد های PHP و در حقیقت تابع mysqli_connect() استفاده میشود.

نحوه نوشتن ارتباط با سرور MYSQL : این تابع شامل بخش های مختلفی است که نوشتن همه آن ها به صورت اختیاری است. در این بخش به معرفی مهمترین این بخش های میپردازیم. mysqli_connect(host,username,password,dbname);

پارامتر	توضیحات
هاست	نوشتن آن اختیاری است. چه نام هاست و یا IP
نام کاربری	نوشتن آن اختیاری است. نام کاربری در MYSQL
کلمه عبور	نوشتن آن اختیاری است.
dbname	نوشتن آن اختیاری است. به طور پیش فرض نامی برایش ساخته میشود.

در مثال زیر ارتباط را در متغیر $con ذخیره میکنیم و با استفاده از تابع mysqli_connect() ارتباط با سرور برقرار شده است. حال باید این ارتباط مورد بررسی قرار گیرد که هیچ خطایی نداشته

باشد. تابع mysqli_connect_errno() آخرین کد خطایی را که تابع mysqli_connect() دارد را نمایش میدهد.

```
<?php
// Create connection
$con=mysqli_connect("example.com","peter","abc123","my_db");

// Check connection
if (mysqli_connect_errno($con))
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
?>
```

بستن ارتباط : به صورت خود کار این ارتباط در پایان اجرای اسکریپت های پی اچ پی پایان می یابد اما اگر بخواهید خودتان این ارتباط را زود تر ببندید از تابع mysqli_close() میتوانید استفاده کنید. به مثال زیر دقت کنید. این مثال همان شرایط مثال قبل را داراست با این تفاوت که قبل از پایان تابع این ارتباط بسته میشود.

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
 // Check connection
if (mysqli_connect_errno($con))
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
 mysqli_close($con);
?>
```

دستور insert into : ازاین دستور برای اضافه کردن اطلاعات جدید در جدول پایگاه داده استفاده میشود. این دستور هم همانند دستور هایی دیگری که آموزش دادیم دارای یک سری قوانین برای نوشتن آن در PHP است. میتوان دستور insert into را به دو صورت نوشت.

روش اول: در این روش نمیتوان نام یک ستون خاص که درآن اطلاعات مورد نظر نوشته شده را به جدول پایگاه داده اضافه کرد، تنها میتوان مقادیر و ارزش های داده های مورد نظر را به صورت تک تک وارد نمود.

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)
```

روش دوم: در این روش میتوان نام ستون ها و ارزش هر بخش را وارد نمود

```
1  INSERT INTO table_name (column1, column2, column3,...)
2  VALUES (value1, value2, value3,...)
```

همانند تمام بخش های پایگاه داده، از تابع mysqli_query() برای اجرا شدن PHP استفاده میکنیم. این تابع query یا کامنت ها به پایگاه داده ارسال میکند.

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
 mysqli_query($con,"INSERT INTO Persons (FirstName, LastName, Age)
VALUES ('Peter', 'Griffin',35)");
 mysqli_query($con,"INSERT INTO Persons (FirstName, LastName, Age)
VALUES ('Glenn', 'Quagmire',33)");
 mysqli_close($con);
?>
```

همان طور که در مثال مشاهده می کنید، اطلاعات دو نفر به نام های Peter و Glenn اضافه نمودیم. در این مثال از روش اول وارد کردن اطلاعات استفاده شد.

حال میخواهیم از روش دوم برای وارد کردن اطلاعات در جدول پایگاه داده استفاده کنیم. برای این کار نیازمند داشتن ستون و اطلاعات درون ان هستیم. به همین دلیل یک فرم HTML همانند فرم هایی که در مباحث فرم ایجاد کرده بودیم، میسازیم.

فرم HTML:

```
<html>
<body>
 <form action="insert.php" method="post">
Firstname: <input type="text" name="firstname">
Lastname: <input type="text" name="lastname">
Age: <input type="text" name="age">
<input type="submit">
</form>
 </body>
</html>
```

در مثال زیر هنگامی که کابر دکمه ارسال را میزند اطلاعات درون فرم به بخش insert.php فرستاده میشود، سپس به پایگاه داده ارتباط پیدا میکند و ارزش ها را که در متغیر $-post نوشته شده است را بررسی میکند. تابع mysqli_query() هر آنچه که در دستور insert into نوشته میشود را اجرا کرده و رکورد های جدید را به جدول "Persons" اضافه میکند.

مثال برای روش دوم استفاده از insert into

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
```

```
$sql="INSERT INTO Persons (FirstName, LastName, Age)
VALUES
('$_POST[firstname]','$_POST[lastname]','$_POST[age]')";

if (!mysqli_query($con,$sql))
  {
  die('Error: ' . mysqli_error($con));
  }
echo "1 record added";

mysqli_close($con);
?>
```

دستور select : میخواهیم بررسی کنیم چگونه میتوان این اطلاعات را از جدول انتخاب کرد. برای این منظور از دستور select استفاده میکنیم. روش نوشتن این دستور به صورت زیر است:

```
SELECT column_name(s)
FROM table_name
```

در مثال زیر میخواهیم داده های ذخیره شده را انتخاب کنیم. با استفاده از کاراکتر * و دستور select میتوان تمام داده های موجود در جدول را انتخاب کرد.

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
 $result = mysqli_query($con,"SELECT * FROM Persons");
```

در این مثال داده های ذخیره شده در متغیر $result توسط تابع mysqli_query() بازگردانی میشوند فقط باید توجه داشت که در این مثال فقط اطلاعات را استخراج میکند اما ترتیبی برای آن

```
while($row = mysqli_fetch_array($result))
 {
 echo $row['FirstName'] . " " . $row['LastName'];
 echo "<br>";
 }
mysqli_close($con);
?>
```

ندارد و مرتب نمایش نمیدهد.
خروجی این مثال:

Peter Griffin
Glenn Quagmire

در مثال بعدی با استفاده از تابع mysqli_fetch_array() میتوان ردیف اول اطلاعات ذخیره شده را انتخاب کرد. هر چه در این تابع فراخوان شود به ترتیب ردیف نمایش داده میشود. تابع mysqli_fetch_array() به صورت یک حلقه عمل میکند. تا زمانی که ردیفی برای نمایش دادن وجود داشته باشد این تابع نمایش میدهد. برای نمابش داده های هر ردیف از متغیر $row در PHP استفاده میکنیم. داده های استخراجی در یک جدول نمایش داده میشوند.

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
// Check connection
if (mysqli_connect_errno())
 {
 echo "Failed to connect to MySQL: " . mysqli_connect_error();
 }
$result = mysqli_query($con,"SELECT * FROM Persons");

echo "<table border='1'>
<tr>
<th>Firstname</th>
<th>Lastname</th>
</tr>";
while($row = mysqli_fetch_array($result))
 {
```

```
  echo "<tr>";
  echo "<td>" . $row['FirstName'] . "</td>";
  echo "<td>" . $row['LastName'] . "</td>";
  echo "</tr>";
  }
echo "</table>";
 mysqli_close($con);
?>
```

خروجی این دستور به صورت زیر است:

Firstname	Lastname
Glenn	Quagmire
Peter	Griffin

نحوه نوشتن شرط where در پایگاه داده

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

این کد های نشان میدهد که چه متغیری select میشود و از کدام جدول و در این بخش شرط where مشخص میکند که کدام بخش خاص انتخاب شود.

ابتدا به پایگاه داده متصل شدیم، وجود خطا در اتصال را بررسی کردیم و تا زمانی که ردیفی باقی باشد شرط انتخاب نام ” peter “ را بررسی کرده و نمایش میدهد.

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
 $result = mysqli_query($con,"SELECT * FROM Persons
WHERE FirstName='Peter'");
 while($row = mysqli_fetch_array($result))
  {
  echo $row['FirstName'] . " " . $row['LastName'];
  echo "<br>";
```

```
 }
?>
```

دستور update : در تمام جداول باید این امکان وجود داشته باشد که در صورت نیاز بتوان اطلاعات درون جدول را تغییر داد و یا به روز رسانی کرد. برای به روز رسانی داده های جداول ساخته شده در پایگاه داده از دستور update استفاده میکنیم. با استفاده از دستور Update داده های تغییر میکنند اما این که کدام داده در کدام جدول تغییر کند باید با دستور where مشخص شود. در صورتی که دستور where را ننویسید تغییرات بر روی تمام جداول ایجاد خواهد شد.

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

در این مثال میخواهیم سن فردی به نام peter را تغییر دهیم.

ابتدا به پایگاه داده متصل می شویم و بررسی میکنیم که خطایی در ایجاد این ارتباط وجود نداشته باشد سپس اطلاعات مورد نیاز را در تابع mysqli_query() تعریف میکنیم. میخواهیم در جدول "pesron" سن برابر با ۳۶ شود. با استفاده از دستور where جزئیات فیلد مربوطه را ذکر میکنیم.

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
 mysqli_query($con,"UPDATE Persons SET Age=36
WHERE FirstName='Peter' AND LastName='Griffin'");
 mysqli_close($con);
?>
```

دستور Delete : هنگامی که بخواهیم در MYSQL یک یا چند رکورد را از جدولی حذف کنیم از دستور Delete استفاده میکنیم. همانند دستور Update که برای به روز رسانی داده ها استفاده میشود، در دستور Delete نیز باید از شرط Where استفاده شود. در صورتی که هیچ شرطی را برای جدول در نظر نگیرید تمام رکورد ها با استفاده از دستور Delete حذف میشوند.

```
DELETE FROM جدول نام
WHERE some_column = some_value
```

در مقابل دستور DELETE FROM نام جدولی که میخواهیم در رکورد هایش را تغییر دهیم را نوشته و با استفاده از شرط WHERE مشخص میکنیم که چه رکورد هایی حذف شوند.

با استفاده از یک مثال نحوه استفاده از این دستور را بهتر متوجه شوید. میخواهیم در جدول 'person' رکوردی را حذف کنیم، برای این که تمام اطلاعات حذف نشوند باید برای ان شرط قرار دهیم، شرط این مثال حذف نام خانوادگی Griffin است.

ابتدا به پایگاه داده متصل می شویم و بررسی میکنیم که خطایی در ایجاد این ارتباط وجود نداشته باشد سپس اطلاعات مورد نیاز را در تابع mysqli_query() تعریف میکنیم. سپس در جدول persons نام خانوادگی Griffin حذف میشود

```
<?php
$con=mysqli_connect("example.com","peter","abc123","my_db");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
 mysqli_query($con,"DELETE FROM Persons WHERE LastName='Griffin'");
 mysqli_close($con);
?>
```

نمونه فایل ایجاد پایگاه داده :

```
<?php
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('could not connect:'.mysql_error());
}
if(mysql_query("CREATE DATABASE testbank",$con))
{
 echo"created";
 }
Else
{

echo"error creating
database.".mysql_error();
 }
mysql_close($con);
?>
```

نمونه فایل ایجاد جدول :

```
<?php
$con=mysql_connect("localhost","root","");
if(!$con)
{
die("no connect");
}
mysql_select_db("testbank",$con);

$query="create table tblUser
      (name varchar(15),
       family varchar(20),
       age int)";

mysql_query($query,$con);

mysql_close($con);
?>
```

نمونه فایل درج داده در جدول :

فایل php برای ذخیره کردن داده ها

```
<?php
$con=mysql_connect("localhost","root","");
mysql_select_db("testbank",$con);

//INSERT INTO TableName (Field1,Field2,...)
VALUES('value1','value2',...)
$query="INSERT INTO tblUser (name,family,age) VALUES
('$_POST[txtName]','$_POST[txtFamily]','$_POST[txtAge]')";

mysql_query($query);

mysql_close($con);

echo "Inserted";
?>
```

فایل فرم html

```
<html>
<body>
<form action="insert.php"
method="post">
Name:<input type="text"
name="txtName"/><br/>
Family:<input type="text"
name="txtFamily"/><br/>
Age:<input type="text"
name="txtAge"/><br/>
<input type="submit"/>
<br/>
</form>
</body>
</html>
```

نمونه فایل تغییر داده ها در جدول :

فایل php برای جستجوی داده ها جهت اعمال تغییرات

```
<?php
session_start();
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('could not connect:'.mysql_error());
}
mysql_SELECT_db("testbank",$con);
$result=mysql_query("SELECT * FROM
tblUser where name='$_POST[txtName]'");
echo"<table border='1'>
<tr>
<th> name</th>
<th> family</th>
<th> age</th>
</tr>";
while ($row=mysql_fetch_array($result))
{
echo"<tr>";
echo"<td>".$row['name']."</td>";
echo"<td>".$row['family']."</td>";
echo"<td>".$row['age']."</td>";
echo"</tr>";
}
echo"</table>";
mysql_close($con);

//session_start();
$_SESSION["txtname2"]=$_POST[txtName];
$session_data=session_encode();

?>
<form action="update.php" method="post">
name:<input type="text"
name="txtname1"/><br/>
Family:<input type="text"
name="txtfamily1"/><br/>
Age:<input type="text"
name="txtage1"/><br/>
<input type="submit"/>
</form>
```

فایل فرم html

```
<html>
<body>
<form action="update1.php" method="post">
Name:<input type="text" name="txtName"/><br/>
<input type="submit"/>
<br/>
</form>
</body>
</html>
```

فایل php برای انجام تغییرات

```
<?php
$con=mysql_connect("localhost","root","");
mysql_select_db("testbank",$con);
session_decode($session_data);
$s1=$_SESSION["txtname2"];

$query="UPDATE TblUser SET
name='$_POST[txtname1]',family='$_POST[txtfamily
1]',age='$_POST[txtage1]' WHERE (name='$s1')";
$x=mysql_query($query);

if($x)
    echo "Updated";
else echo "Not found";
mysql_close($con);
?>
<a href="main.html"> main </a>
```

نمونه فایل نمایش لیست محتویات جدول :

```
<?php
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('could not connect:'.mysql_error());
}
mysql_SELECT_db("testbank",$con);
$result=mysql_query("SELECT * FROM tblUser order
by family");
echo"<table border='1'>
<tr>
<th> name</th>
<th> family</th>
<th> age</th>
</tr>";
```

```
while ($row=mysql_fetch_array($result))

{
echo"<tr>";
echo"<td>".$row['name']."</td>";
echo"<td>".$row['family']."</td>";
echo"<td>".$row['age']."</td>";
echo"</tr>";
}
echo"</table>";
mysql_close($con);
?>
```

نمونه فایل حذف داده از جدول :

فایل فرم html

```
<html>
<body>
<form action="delete.php" method="post">
Name:<input type="text" name="txtName"/><br/>
<input type="submit"/>
<br/>
</form>
</body>
</html>
```

فایل php برای حذف داده ها

```
<?php
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('could not connect:'.mysql_error());
}
mysql_SELECT_db("testbank",$con);
$result=mysql_query("delete  FROM tblUser
where name='$_POST[txtName]'");
if($result)
{
echo"deleted";
}
else
{
echo"can not delete";
}
mysql_close($con);

?>
```

فایل تولید تصویر یا کد امنیتی

```
<?
//SecImage.php
session_start();
 if( !isset( $_SESSION['SecImageStr']))
 exit();
 $Str = $_SESSION['SecImageStr'];
 $SecImage = imagecreate( 50, 20);
$BgColor = imagecolorallocate( $SecImage, 255, 150, 220);//Background color
$FrColor = imagecolorallocate( $SecImage, 0, 0, 255);//ForeGround color
$LineColor = imagecolorallocate( $SecImage, 255, 0, 150);//Line color
 for($Index = 0; $Index != 3; $Index++)
{
   $LineDegree = rand(15, 50);
   imageline ( $SecImage, $Index, $LineDegree, ($Index+1) * 20, $Index,
$LineColor );
}
 imagestring ( $SecImage, 4, 5, 1, $Str, $FrColor ) ;
imagejpeg ( $SecImage , '', 100 ) ;
 header("Content-type: image/jpg");
imagedestroy ( $SecImage ) ;
 ?>
```

فایل جایی که می خواهیم استفاده کنیم (فایل فرم)

```
<?
//test.php
session_start();
// Create Security Code ...

   $SecCode = md5(rand(0,9999));
   $SecCode = strtolower($SecCode);
   $_SESSION[‘SecImageStr’]  = strtoupper(substr($SecCode,0,4));
?>
<img src=”SecImage.php”>
```

فایلی که در action فرم می نوسیم برای بررسی درستی یا نادرستی کد

```
<?   //check.php
session_start();
 $UserSecCode = strtolower($_POST['SecCode']);
$SysSecCopde =
strtolower($_SESSION['SecImageStr']);
```

```
if($UserSecCode != $SysSecCopde)
{
   die('The Sec Code is invalid!');
}  ادامه برنامه//
?>
```