

به نام خدا

برنامه نویسی مبتنی بر وب



PHP

PHP چیست؟

- **PHP** زبانی اسکریپتی **سمت سرور** برای ایجاد صفحات وب پویا و تعاملی می باشد که برگرفته از عبارت **Hypertext Preprocessor** است. این زبان به صورت Open Source و رایگان عرضه می شود.

چرا PHP؟

- هسته **WordPress** که بزرگترین سیستم مدیریت محتوا (CMS) می باشد، با زبان PHP نوشته شده است.
- **Facebook** از این زبان استفاده کرده است.
- این زبان بر روی پلت فرم های مختلف اجرا می شود (Windows, Linux, Unix, Mac OS X, etc).
- با اکثر سرورها هم خوانی دارد (Apache, IIS, etc).
- به راحتی از سایت رسمی آن به آدرس <http://www.php.net/> قابل دانلود می باشد.

یک فایل PHP چیست؟

- فایل های PHP می تواند شامل متن، HTML، CSS، JavaScript و کد های PHP باشد. کدهای PHP بر روی سرور اجرا می گردد و نتیجه آن به مرورگر کلاینت از طریق HTML برگشت داده می شود. پسوند فایل های PHP به صورت php می باشد.



Paradigm	Imperative, functional, object-oriented, procedural, reflective
Designed by	Rasmus Lerdorf
Developer	Zend Technologies
First appeared	1995; 22 years ago ^[1]
Stable release	7.2.0 ^[2] / November 30, 2017; 27 days ago
Typing discipline	Dynamic, weak
Implementation language	C (primarily; some components C++)
OS	Unix-like, Windows
License	PHP License (most of Zend engine under Zend Engine License)
Filename extensions	.php, .phtml, .php3, .php4, .php5, .php7, .phps, .php-s
Website	php.net 

نگاہی بہ آمار

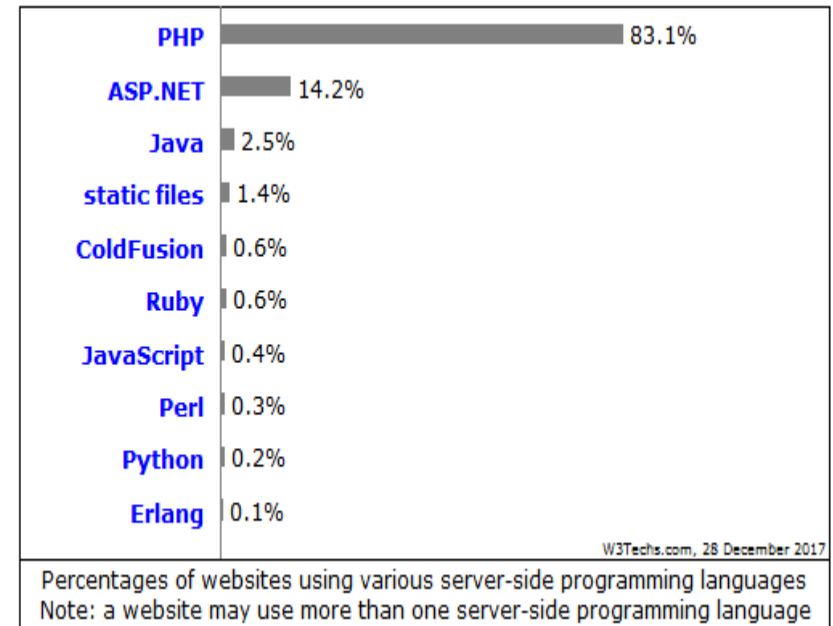
- <https://w3techs.com/>

Server-side Programming Languages

Most popular server-side programming languages























© W3Techs.com	usage	change since 1 November 2017
1. PHP	83.1%	+0.2%
2. ASP.NET	14.2%	-0.1%
3. Java	2.5%	
4. static files	1.4%	-0.1%
5. ColdFusion	0.6%	

percentages of sites



نگاہی بہ آمار

- The 2017 Top Programming Languages
- *IEEE Spectrum*

Language Rank	Types	Spectrum Ranking
1. Python	 	100.0
2. C	  	99.7
3. Java	  	99.5
4. C++	  	97.1
5. C#	  	87.7
6. R		87.7
7. JavaScript	 	85.6
8. PHP		81.2
9. Go	 	75.1
10. Swift	 	73.7

نحوه نصب PHP

- برای استفاده از PHP می توان به دو روش عمل کرد:

❖ داشتن یک هاست وب که قابلیت پشتیبانی از PHP و MySQL را داشته باشد.

✓ در صورت استفاده از این روش تنها کافی است که کدهای php خود را نوشته و سپس فایل های ایجاد شده را در دایرکتوری مورد نظر در هاست خود قرار دهیم.

❖ نصب یک وب سرور بر روی کامپیوتر خود و سپس نصب PHP و MySQL.

✓ نصب وب سرور

✓ نصب PHP

✓ نصب MySQL

➤ در وب سایت رسمی PHP دستورالعمل های لازم برای نصب php وجود دارد.

<http://php.net/manual/en/install.php>

نوشتن کدهای PHP

- در اسکریپت های PHP کدهایی را که می نویسم باید در داخل تگ هایی قرار دهیم. کدها می توانند در هرجایی در سند باشند. کدهای PHP با `<?php` شروع می شوند و با `?>` تمام می شوند.

```
<?php
// PHP code goes here
?>
```

Example

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

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

My first PHP page

Hello World!



Code 1.txt

نکته

- عبارات PHP با ; پایان می پذیرند.

```
<?php  
echo "Hello World!";  
?>
```

نوشتن توضیحات در کدهای PHP

- برای نوشتن توضیحات (Comments) می توان از روش های زیر استفاده کرد.

❖ از `//` برای نوشتن توضیحات تک خطی

❖ از `#` برای نوشتن توضیحات تک خطی

❖ از `/* */` برای نوشتن توضیحات چند خطی

```
<?php
// This is a single-line comment

# This is also a single-line comment

/*
This is a multiple-lines comment block
that spans over multiple
lines
*/

// You can also use comments to leave out parts of a code line
$x = 5 /* + 15 */ + 5;
echo $x;
?>
```

نکته

- در زبان PHP همه کلمات کلیدی (مثل if, else, while, ...), کلاس ها، توابع و تابع های تعریف شده توسط کاربر حساس به حروف کوچک و بزرگ نیست (NOT case-sensitive)، اما نام متغیرها حساس به حروف است.

Example

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

<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>

</body>
</html>
```

Hello World!
Hello World!
Hello World!

Example

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

<?php
$color = "red";
echo "My car is " . $color . "<br>";
echo "My house is " . $COLOR . "<br>";
echo "My boat is " . $coLOR . "<br>";
?>

</body>
</html>
```

My car is red
My house is
My boat is



Code 2.txt

متغیرها در PHP

- در PHP متغیرها با علامت \$ شروع می شوند و به دنبال آن نام متغیر می آید.
نام متغیر باید با یک حرف یا کاراکتر _ (underscore) شروع شود و نباید با رقم شروع شود. همچنین نام متغیرها نسبت به حروف کوچک و بزرگ حساس است.

Example

```
<?php  
$txt = "Hello world!";  
$x = 5;  
$y = 10.5;  
?>
```


متغیرها در PHP

- هنگامی که یک مقدار متنی را به یک متغیر نسبت می دهید، مقدار متنی را باید در داخل گیومه (quotes) قرار داد.

```
$txt = "Hello world!";
```

متغیرها در PHP

- بر خلاف سایر زبان های برنامه نویسی، PHP دستوری برای اعلان متغیر ندارد. متغیر زمانی که اولین مقدار را به آن نسبت می دهید، ایجاد می شود.

Example

```
<?php  
$txt = "Hello world!";  
$x = 5;  
$y = 10.5;  
?>
```

دستور echo

- دستور `echo` برای مشاهده خروجی در PHP بکار می رود.

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

```
<?php
$txt = "W3Schools.com";
echo "I love $txt!";
?>
```

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



Code 3.txt

I love W3Schools.com!

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

```
<?php
$txt = "W3Schools.com";
echo "I love " . $txt . "!";
?>
```

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



Code 4.txt

مثال

- اسکرپت زیر جمع دو متغیر را نمایش می دهد.

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

<?php
$x = 5;
$y = 4;
echo $x + $y;
?>

</body>
</html>
```



Code 5.txt

PHP is a Loosely Typed Language

- در زبان های Loosely Typed نیازی به تعریف متغیرها و تعیین نوع داده آنها نمی باشد. در این زبان ها تغییر نوع داده ها به طور خودکار و بر حسب نیاز توسط خود زبان برنامه سازی انجام می شود. زبان **PHP** از جمله این زبان ها است و به طور اتوماتیک متغیرها را به نوع داده درست بر حسب مقدارش تبدیل می کند.

حوزه (Scope) متغیرها در PHP

- در PHP متغیرها هر جایی در اسکریپت می توانند تعریف شوند. منظور از **حوزه یک متغیر** قسمتی از اسکریپت است که متغیر در آن می تواند مورد استفاده یا اشاره قرار گیرد. PHP دارای سه حوزه ی مختلف متغیر است که عبارتند از:

local ✓

global ✓

static ✓

متغیر سراسری Global Scope

- متغیری که در خارج از یک تابع تعریف شده است حوزه اش سراسری (Global) است و تنها در خارج از تابع قابل دستیابی است.

Example

```
<?php
$x = 5; // global scope

function myTest() {
    // using x inside this function will generate an error
    echo "<p>Variable x inside function is: $x</p>";
}

myTest();

echo "<p>Variable x outside function is: $x</p>";
?>
```

Variable x inside function is:

Variable x outside function is: 5



Code 6.txt

متغیر محلی Local Scope

- متغیری که در داخل یک تابع تعریف شده باشد، حوزه اش محلی (Local) است و تنها در داخل آن تابع قابل دستیابی است.

Example

```
<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>
```

Variable x inside function is: 5

Variable x outside function is:



Code 7.txt

نکته

- می توان متغیرهای محلی با نام های یکسان در توابع مختلف داشت، زیرا متغیرهای محلی تنها در همان تابعی که اعلان شده اند، شناسایی می شوند و قابل دسترسی هستند.

استفاده از کلمه کلیدی global

- کلمه کلیدی `global` به منظور دستیابی به یک متغیر سراسری در داخل یک تابع بکار می رود. برای این منظور کلمه `global` را قبل از نام متغیر در تابع ذکر می کنیم.

Example

```
<?php
$x = 5;
$y = 10;

function myTest() {
    global $x, $y;
    $y = $x + $y;
}

myTest();
echo $y; // outputs 15
?>
```

15



Code &txt

PHP also stores all global variables in an array called `$GLOBALS[index]`. The *index* holds the name of the variable. This array is also accessible from within functions and can be used to update global variables directly.

The example above can be rewritten like this:

Example

```
<?php
$x = 5;
$y = 10;

function myTest() {
    $GLOBALS['y'] = $GLOBALS['x'] + $GLOBALS['y'];
}

myTest();
echo $y; // outputs 15
?>
```

استفاده از کلمه کلیدی static

- به صورت عادی، وقتی یک تابع اجرا می شود، تمام متغیرهای آن از بین می روند. گاهی مواقع ما نیاز داریم که متغیرهای محلی تعریف شده در داخل تابع پس از اجرا تابع از بین نروند. برای این منظور باید کلمه کلیدی `static` را زمان اعلان متغیر استفاده کنید. در متغیرها از نوع استاتیک متغیر تنها یکبار مقدار دهی اولیه می شود. با اتمام اسکریپت، متغیرهای محلی استاتیک از بین می روند.

مثال

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
function myTest() {
```

```
    static $x = 0;
```

```
    echo $x;
```

```
    $x++;
```

```
}
```

```
myTest();
```

```
echo "<br>";
```

```
myTest();
```

```
echo "<br>";
```

```
myTest();
```

```
?>
```

```
</body>
```

```
</html>
```

0

1

2



Code 9.txt

دستور echo و print

- این دو دستور کم و بیش یکسان می باشند و هر دو برای نمایش خروجی بکار می روند. تفاوت این دو دستور خیلی کم و به صورت زیر است.
- ✓ دستور echo هیچ مقدار برگشتی ندارد ولی دستور print مقدار 1 را برگشت می دهد بنابراین می توان از این دستور در عبارات استفاده کرد.
- ✓ دستور echo می تواند چندین پارامتر بگیرد (اگر چه به ندرت بکار می رود) ولی دستور print تنها می تواند یک آرگومان را بگیرد.
- ✓ دستور echo به مراتب سریعتر از دستور print است.
- ✓ در دستور echo برای اتصال دو رشته هم می توان از "." و هم از "," استفاده کرد.
- ✓ در دستور print برای اتصال دو رشته فقط می توان از "." استفاده کرد.

نکته

- دستور echo و print می توانند با پرانتز یا بدون پرانتز بکار روند.
- echo or echo()
- print or print()

مثالی از دستور echo

DisplayText

Example

```
<?php  
echo "<h2>PHP is Fun!</h2>";  
echo "Hello world!<br>";  
echo "I'm about to learn PHP!<br>";  
echo "This ", "string ", "was ", "made ", "with multiple parameters."  
?>
```

PHP is Fun!

Hello world!

I'm about to learn PHP!

This string was made with multiple parameters.



Code 10.txt

مثالی از دستور echo

Display Variables

Example

```
<?php
$txt1 = "Learn PHP";
$txt2 = "W3Schools.com";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP at " . $txt2 . "<br>";
echo $x + $y;
?>
```

Learn PHP

Study PHP at W3Schools.com
9



Code 11.txt

مثال از دستور print

Example

```
<?php  
print "<h2>PHP is Fun!</h2>";  
print "Hello world!<br>";  
print "I'm about to learn PHP!";  
?>
```

PHP is Fun!

Hello world!

I'm about to learn PHP!

مثال از دستور print

Example

```
<?php
$txt1 = "Learn PHP";
$txt2 = "W3Schools.com";
$x = 5;
$y = 4;

print "<h2>" . $txt1 . "</h2>";
print "Study PHP at " . $txt2 . "<br>";
print $x + $y;
?>
```

Learn PHP

Study PHP at W3Schools.com

9

نوع های داده PHP Data Types

• PHP از نوع های داده زیر پشتیبانی می کند:

- String
- Integer
- Float (floating point numbers - also called double)
- Boolean
- Array
- Object
- NULL
- Resource

نوع داده String

- یک رشته (string) دنباله ای از کاراکترها (characters) می باشد. یک رشته می تواند هر متنی داخل گیومه (تکی یا دوتایی single or double quotes) باشد.

Example

```
<?php
$x = "Hello world!";
$y = 'Hello world!';

echo $x;
echo "<br>";
echo $y;
?>
```

Hello world!
Hello world!



Code 14.txt

نوع داده Integer

• نوع داده صحیح یک عدد غیر اعشاری بین $-2,147,483,648$ و $2,147,483,647$ است.

✓ باید حداقل یک رقم داشته باشد.

✓ اعشار نداشته باشد.

✓ می تواند مثبت یا منفی باشد.

✓ می تواند در فرمت های مبنای ۱۰، مبنای ۱۶ (با پیشوند 0x) یا مبنای ۸ باشد (با پیشوند 0).

نکته

- تابع `var_dump()` در PHP برای تعیین نوع داده و مقدار بکار می رود.

Example

```
<?php  
$x = 5985;  
var_dump($x);  
?>
```

`int(5985)`



Code 15.txt

نوع داده Float

- این نوع برای اعداد اعشاری بکار می رود.

Example

```
<?php  
$x = 10.365;  
var_dump($x);  
?>
```

float(10.365)



Code 16.txt

نوع داده Boolean

- نوع Boolean دو حالت ممکن را نمایش می دهد: **TRUE** یا **FALSE**
- این نوع داده اغلب در تست های شرطی (conditional testing) بکار می رود.

```
$x = true;  
$y = false;
```

آرایه Array

- آرایه مقادیر چندگانه را در یک متغیر ذخیره می کند.

Example

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

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

- تابع `var_dump()` مقادیر و نوع آرایه را بر می گرداند.

نوع Object

- object نوع داده ای است که داده ها و اطلاعات در مورد نحوه پردازش آن داده را ذخیره می کند. در PHP یک شی باید به صورت صریح تعریف شود. در ابتدا باید کلاس شی تعریف شود. برای این منظور از کلمه کلیدی `class` استفاده می شود که می تواند شامل **خصوصیت ها** (`properties`) و **متدها** (`methods`) باشد.

مثال نوع Object

Example

```
<?php
class Car {
    function Car() {
        $this->model = "VW";
    }
}

// create an object
$herbie = new Car();

// show object properties
echo $herbie->model;
?>
```

مقدار تهی NULL Value

- Null یک نوع داده ویژه است که تنها می تواند یک مقدار داشته باشد: Null
- یک متغیر از نوع Null، متغیری است که هیچ مقداری به آن نسبت داده نشده است.
- نکته: اگر متغیری بدون مقدار ایجاد شود، به طور اتوماتیک مقدار Null به آن نسبت داده می شود.

Example

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

NULL



Code 17.txt

نوع Resource

- نوع ویژه resource، نوع داده واقعی ندارد و برای ذخیره یک رفرنس به توابع و منابع خارجی بکار می رود. یک مورد رایج استفاده از این نوع داده در فراخوانی پایگاه داده ها است.

توابع رشته PHP String Functions

- طول یک رشته (Length of a String)
- شمارش تعداد کلمات در یک رشته (Count The Number of Words in a String)
- معکوس کردن رشته (Reverse a String)
- جستجوی یک متن خاص در یک رشته (Search For a Specific Text Within a String)
- جایگزینی متن با یک رشته (Replace Text Within a String)

به دست آوردن طول یک رشته

- تابع strlen() طول رشته را بر می گرداند.

Example

```
<?php  
echo strlen("Hello world!"); // outputs 12  
?>
```



Code 18.txt

شمارش تعداد کلمات در یک رشته

- تابع `str_word_count()` تعداد کلمات در یک رشته را شمارش می کند.

Example

```
<?php  
echo str_word_count("Hello world!"); // outputs 2  
?>
```



Code 19.txt

معکوس کردن یک رشته

- تابع `strrev()` رشته مورد نظر را معکوس می کند.

Example

```
<?php  
echo strrev("Hello world!"); // outputs !dlrow olleH  
?>
```



Code 20.txt

جستجوی یک متن خاص در رشته

- تابع `strpos()` یک متن خاص را در یک رشته جستجو می کند. اگر متن مورد نظر در رشته یافت شود، تابع مکان اولین کاراکتر متن مورد نظر را در رشته، بر می گرداند.
- توجه: مکان اولین کاراکتر در رشته صفر می باشد (نه یک).

Example

```
<?php  
echo strpos("Hello world!", "world"); // outputs 6  
?>
```



Code 21.txt

جایگزینی متن در یک رشته

- تابع `str_replace()` تعدادی کاراکتر را با تعدادی کاراکتر دیگر در رشته جایگزین می کند.

Example

```
<?php  
echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!  
?>
```



Code 22.txt

رفرنس کامل توابع رشته ای در PHP

- برای مشاهده رفرنس کامل توابع رشته ای در PHP می توانید به آدرس زیر مراجعه کنید.

https://www.w3schools.com/php/php_ref_string.asp

ثابت Constants

- ثابت یک شناسه (نام) برای یک مقدار ساده می باشد. مقدار ثابت در طول اجرای اسکریپت قابل تغییر نیست. به طور پیش فرض نام ثابت ها حساس به حروف کوچک و بزرگ می باشد. نام ثابت با یک حرف یا کاراکتر _ شروع می شود و در ادامه می توان حروف و ارقام و کاراکتر _ را بکار برد. علامت \$ در ابتدای نام ثابت نیاز نیست. به طور کلی ثابت ها محدوده سراسری دارند و در همه جای یک اسکریپت قابل استفاده می باشند.

نحوه ایجاد ثابت

- برای ایجاد ثابت از تابع `define()` استفاده می شود.

```
define(name, value, case-insensitive)
```

Parameters:

- *name*: Specifies the name of the constant
- *value*: Specifies the value of the constant
- *case-insensitive*: Specifies whether the constant name should be case-insensitive. Default is false

مثال

- مثالی از تعریف یک ثابت با نام حساس به حروف کوچک و بزرگ (case-sensitive).

Example

```
<?php  
define("GREETING", "Welcome to W3Schools.com!");  
echo GREETING;  
?>
```

Welcome to W3Schools.com!



code 23.txt

مثال

- مثالی از تعریف یک ثابت با نام غیر حساس به حروف کوچک و بزرگ (case-insensitive).

Example

```
<?php  
define("GREETING", "Welcome to W3Schools.com!", true);  
echo greeting;  
?>
```



Code 24.txt

سراسری بودن ثابت Constants are Global

- ثوابت به طور اتوماتیک سراسری (global) هستند و می توانند در تمام اسکریپت استفاده شوند.

✓ مثال زیر نحوه استفاده از یک ثابت در داخل یک تابع را نشان می دهد، حتی با اینکه آن ثابت در خارج تابع تعریف شده است.

Example

```
<?php
define("GREETING", "Welcome to W3Schools.com!");

function myTest() {
    echo GREETING;
}

myTest();
?>
```

Welcome to W3Schools.com!

عملگرها Operators

- عملگرها برای اجرای عملیاتی بر روی مقادیر و متغیرها بکار می روند. در PHP عملگرها به گروه های زیر تقسیم می شوند.

- ❖ عملگرهای محاسباتی (Arithmetic operators)
- ❖ عملگرهای تخصیص (Assignment operators)
- ❖ عملگرهای مقایسه ای (Comparison operators)
- ❖ عملگرهای افزایش و کاهش (Increment/Decrement operators)
- ❖ عملگرهای منطقی (Logical operators)
- ❖ عملگرهای رشته ای (String operators)
- ❖ عملگرهای آرایه (Array operators)

عملگرهای محاسباتی Arithmetic

Operator	Name	Example	Result
+	Addition	$\$x + \y	Sum of $\$x$ and $\$y$
-	Subtraction	$\$x - \y	Difference of $\$x$ and $\$y$
*	Multiplication	$\$x * \y	Product of $\$x$ and $\$y$
/	Division	$\$x / \y	Quotient of $\$x$ and $\$y$
%	Modulus	$\$x \% \y	Remainder of $\$x$ divided by $\$y$
**	Exponentiation	$\$x ** \y	Result of raising $\$x$ to the $\$y$ 'th power (Introduced in PHP 5.6)

مثال

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

```
<?php  
$x = 10;  
$y = 6;
```

```
echo $x + $y;  
?>
```

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

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

```
<?php  
$x = 10;  
$y = 6;
```

```
echo $x % $y;  
?>
```

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



Code 25.txt

عملگرهای تخصیص Assignment

Assignment	Same as...	Description
$x = y$	$x = y$	The left operand gets set to the value of the expression on the right
$x += y$	$x = x + y$	Addition
$x -= y$	$x = x - y$	Subtraction
$x *= y$	$x = x * y$	Multiplication
$x /= y$	$x = x / y$	Division
$x \%= y$	$x = x \% y$	Modulus

عملگرهای مقایسه Comparison

Operator	Name	Example	Result
==	Equal	<code>\$x == \$y</code>	Returns true if <code>\$x</code> is equal to <code>\$y</code>
===	Identical	<code>\$x === \$y</code>	Returns true if <code>\$x</code> is equal to <code>\$y</code> , and they are of the same type
!=	Not equal	<code>\$x != \$y</code>	Returns true if <code>\$x</code> is not equal to <code>\$y</code>
<>	Not equal	<code>\$x <> \$y</code>	Returns true if <code>\$x</code> is not equal to <code>\$y</code>
!==	Not identical	<code>\$x !== \$y</code>	Returns true if <code>\$x</code> is not equal to <code>\$y</code> , or they are not of the same type
>	Greater than	<code>\$x > \$y</code>	Returns true if <code>\$x</code> is greater than <code>\$y</code>
<	Less than	<code>\$x < \$y</code>	Returns true if <code>\$x</code> is less than <code>\$y</code>
>=	Greater than or equal to	<code>\$x >= \$y</code>	Returns true if <code>\$x</code> is greater than or equal to <code>\$y</code>
<=	Less than or equal to	<code>\$x <= \$y</code>	Returns true if <code>\$x</code> is less than or equal to <code>\$y</code>

مثال

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

```
<?php
$x = 100;
$y = "100";
```

```
var_dump($x == $y); // returns true because values are equal
?>
```

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

bool(true)



Code 26.txt

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

```
<?php
$x = 100;
$y = "100";
```

```
var_dump($x === $y); // returns false because types are not equal
?>
```

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

bool(false)

عملگرهای افزایش / کاهش Increment / Decrement

Operator	Name	Description
<code>++\$x</code>	Pre-increment	Increments <code>\$x</code> by one, then returns <code>\$x</code>
<code>\$x++</code>	Post-increment	Returns <code>\$x</code> , then increments <code>\$x</code> by one
<code>--\$x</code>	Pre-decrement	Decrements <code>\$x</code> by one, then returns <code>\$x</code>
<code>\$x--</code>	Post-decrement	Returns <code>\$x</code> , then decrements <code>\$x</code> by one

مثال

```
<?php  
$x = 10;  
echo ++$x;  
?>
```

11

```
<?php  
$x = 10;  
echo $x++;  
?>
```

10

```
<?php  
$x = 10;  
echo --$x;  
?>
```

9

```
<?php  
$x = 10;  
echo $x--;  
?>
```

10

عملگرهای منطقی Logical

The PHP logical operators are used to combine conditional statements.

Operator	Name	Example	Result
and	And	<code>\$x and \$y</code>	True if both <code>\$x</code> and <code>\$y</code> are true
or	Or	<code>\$x or \$y</code>	True if either <code>\$x</code> or <code>\$y</code> is true
xor	Xor	<code>\$x xor \$y</code>	True if either <code>\$x</code> or <code>\$y</code> is true, but not both
<code>&&</code>	And	<code>\$x && \$y</code>	True if both <code>\$x</code> and <code>\$y</code> are true
<code> </code>	Or	<code>\$x \$y</code>	True if either <code>\$x</code> or <code>\$y</code> is true
<code>!</code>	Not	<code>!\$x</code>	True if <code>\$x</code> is not true

عملگرهای رشته ای String

PHP has two operators that are specially designed for strings.

Operator	Name	Example	Result
.	Concatenation	\$txt1 . \$txt2	Concatenation of \$txt1 and \$txt2
.=	Concatenation assignment	\$txt1 .= \$txt2	Appends \$txt2 to \$txt1

مثال

```
<?php  
$txt1 = "Hello";  
$txt2 = " world!";  
echo $txt1 . $txt2;  
?>
```

Hello world!

```
<?php  
$txt1 = "Hello";  
$txt2 = " world!";  
$txt1 .= $txt2;  
echo $txt1;  
?>
```

Hello world!

عملگرهای آرایه Array

The PHP array operators are used to compare arrays.

Operator	Name	Example	Result
+	Union	<code>\$x + \$y</code>	Union of <code>\$x</code> and <code>\$y</code>
<code>==</code>	Equality	<code>\$x == \$y</code>	Returns true if <code>\$x</code> and <code>\$y</code> have the same key/value pairs
<code>===</code>	Identity	<code>\$x === \$y</code>	Returns true if <code>\$x</code> and <code>\$y</code> have the same key/value pairs in the same order and of the same types
<code>!=</code>	Inequality	<code>\$x != \$y</code>	Returns true if <code>\$x</code> is not equal to <code>\$y</code>
<code><></code>	Inequality	<code>\$x <> \$y</code>	Returns true if <code>\$x</code> is not equal to <code>\$y</code>
<code>!==</code>	Non-identity	<code>\$x !== \$y</code>	Returns true if <code>\$x</code> is not identical to <code>\$y</code>

ساختارهای تصمیم شرطی در PHP

• در PHP ساختارهای تصمیم شرطی زیر وجود دارد.

- **if statement** - executes some code if one condition is true
- **if...else statement** - executes some code if a condition is true and another code if that condition is false
- **if...elseif....else statement** - executes different codes for more than two conditions
- **switch statement** - selects one of many blocks of code to be executed

ساختار if

- در این ساختار **در صورت درست بودن شرط**، عبارت های نوشته شده در داخل بدنه if اجرا می شود.

Syntax

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

Example

```
<?php
$t = date("H");

if ($t < "20") {
    echo "Have a good day!";
}
?>
```



Code 28.txt

ساختار if...else

- در این ساختار در صورت صحیح بودن شرط دستورات بدنه if و در صورت عدم برقراری شرط دستورات بدنه else، اجرا می شود.

Syntax

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

Example

```
<?php
$t = date("H");

if ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```



Code 29.txt

ساختار if...elseif....else

- این ساختار این امکان را فراهم می کند تا در صورت عدم برقرای شرط دستور if، شرط های دیگری را بررسی نمائیم.

Syntax

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

Example

```
<?php
$t = date("H");

if ($t < "10") {
    echo "Have a good morning!";
} elseif ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```



Code 30.txt

ساختار دستور switch

- دستور `switch` برای اجرای دستورات مختلف بر حسب شرط های متفاوت استفاده می شود.

Syntax

```
switch (n) {  
    case label1:  
        code to be executed if n=label1;  
        break;  
    case label2:  
        code to be executed if n=label2;  
        break;  
    case label3:  
        code to be executed if n=label3;  
        break;  
    ...  
    default:  
        code to be executed if n is different from all labels;  
}
```

Example

```
<?php
$favcolor = "red";

switch ($favcolor) {
    case "red":
        echo "Your favorite color is red!";
        break;
    case "blue":
        echo "Your favorite color is blue!";
        break;
    case "green":
        echo "Your favorite color is green!";
        break;
    default:
        echo "Your favorite color is neither red, blue, nor green!";
}
?>
```



Code 31.txt

while Loops

• حلقه ها از عناصر اصلی برنامه نویسی می باشند. با کمک حلقه ها می توانیم مجموعه ای از دستورات را به دفعات مشخص یا برقراری شرط خاصی اجرا نمائیم.

- **while** - loops through a block of code as long as the specified condition is true
- **do...while** - loops through a block of code once, and then repeats the loop as long as the specified condition is true
- **for** - loops through a block of code a specified number of times
- **foreach** - loops through a block of code for each element in an array

حلقه while

- در ساختار حلقه `while` از یک شرط استفاده می شود و تا برقرار بودن این شرط، مجموعه دستورات عمل های حلقه تکرار می شود.

Syntax

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

Example

```
<?php
$x = 1;

while($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
?>
```

The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5



Code 32.txt

حلقه do...while

- در این ساختار شرط حلقه در انتهای بدنه حلقه چک می شود.

Syntax

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

Example

```
<?php
$x = 1;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```

The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5



Code 33.txt

نکته

- چون در ساختار do while شرط حلقه در انتها چک می شود پس حلقه حداقل یکبار اجرا می شود.

Example

```
<?php
$x = 6;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```

The number is: 6



Code 34.txt

حلقه for

- از این حلقه بیشتر زمانی استفاده می کنیم که تعداد دفعات تکرار حلقه را بدانیم.

Syntax

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

Parameters:

- *init counter*: Initialize the loop counter value
- *test counter*: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
- *increment counter*: Increases the loop counter value

Example

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
?>
```

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
The number is: 6
The number is: 7
The number is: 8
The number is: 9
The number is: 10



Code 35.txt

حلقه foreach

- این حلقه برای کار با آرایه ها استفاده می شود.

Syntax

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

✓ در هر تکرار حلقه، مقدار عنصر جاری آرایه به \$value تخصیص داده می شود و اشاره گر آرایه یکی حرکت می کند تا زمانی که به عنصر آخر آرایه برسد.

Example

```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```

red
green
blue
yellow



Code 36.txt

توابع

- قدرت واقعی PHP نشأت گرفته از توابع این زبان است. در این زبان بیش از هزار تابع درونی وجود دارد (built-in). علاوه بر این برنامه نویس هم می تواند توابع مخصوص خود را ایجاد نماید.
- مزیت های توابع:

A function is a block of statements that can be used repeatedly in a program.

A function will not execute immediately when a page loads.

A function will be executed by a call to the function.

نحوه تعریف تابع

- برای تعریف تابع از کلمه کلیدی `function` استفاده می شود، سپس نام و لیست پارامترهای تابع قرار می گیرند. لیست پارامترها با کاما از هم جدا می شوند.

Syntax

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

Note: A function name can start with a letter or underscore (not a number).

Tip: Give the function a name that reflects what the function does!

Function names are NOT case-sensitive.

Example

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

writeMsg(); // call the function
?>
```

Hello world!



Code 37.txt

Example

```
<?php
function familyName($fname) {
    echo "$fname Refsnes.<br>";
}

familyName("Jani");
familyName("Hege");
familyName("Stale");
familyName("Kai Jim");
familyName("Borge");
?>
```

Jani Refsnes.
Hege Refsnes.
Stale Refsnes.
Kai Jim Refsnes.
Borge Refsnes.



Code 38.txt

Example

```
<?php
function familyName($fname, $year) {
    echo "$fname Refsnes. Born in $year <br>";
}

familyName("Hege", "1975");
familyName("Stale", "1978");
familyName("Kai Jim", "1983");
?>
```

Hege Refsnes. Born in 1975
Stale Refsnes. Born in 1978
Kai Jim Refsnes. Born in 1983



Code 39.txt

Default Argument Value

- در PHP هر یک از آرگومان های تابع می توانند یک مقدار اولیه بگیرند. بدین ترتیب این آرگومان ها اختیاری می شوند و در فراخوانی تابع می توانیم آنها را ذکر نکنیم.

Example

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

setHeight(350);
setHeight(); // will use the default value of 50
setHeight(135);
setHeight(80);
?>
```

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



Code 40.txt

بازگشت مقدار از تابع Returning values

- برای برگشت دادن مقدار از تابع پس از اجرای تابع از دستور return استفاده می شود.

Example

```
<?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);
?>
```

```
5 + 10 = 15
7 + 13 = 20
2 + 4 = 6
```



Code 41.txt

آرایه در PHP

- آرایه در واقع امکان ذخیره چندین مقدار را در یک متغیر فراهم می کند.

Example

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";
?>
```

I like Volvo, BMW and Toyota.

نحوه ایجاد آرایه

• در PHP با تابع `array()` می توان یک آرایه ایجاد کرد. در PHP سه نوع آرایه وجود دارد.

✓ آرایه های ایندکس دار

✓ آرایه های انجمنی

✓ آرایه های چند بعدی

- **Indexed arrays** - Arrays with a numeric index
- **Associative arrays** - Arrays with named keys
- **Multidimensional arrays** - Arrays containing one or more arrays

آرایه های ایندکس دار Indexed Arrays

- در این نوع آرایه ها هر یک از عناصر آرایه، بر اساس اندیس مورد دسترسی قرار می گیرند. به طور پیش فرض اندیس آنها از نوع صحیح بوده و اولین عنصر آرایه اندیس صفر را دارد.

The index can be assigned automatically (index always starts at 0), like this:

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

or the index can be assigned manually:

```
$cars[0] = "Volvo";  
$cars[1] = "BMW";  
$cars[2] = "Toyota";
```

Example

```
<?php  
$cars = array("Volvo", "BMW", "Toyota");  
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";  
?>
```

تعیین طول آرایه

- تابع `count()` برای تعیین طول آرایه (تعداد عناصر آرایه) بکار می رود.

Example

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

3

Example

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
$arlength = count($cars);

for($x = 0; $x < $arlength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>
```

Volvo
BMW
Toyota

آرایه های انجمنی Associative Arrays

- در این نوع آرایه ها برای دسترسی به عناصر آرایه کلیدهایی توسط برنامه نویس مشخص می شوند و این کلیدها برای دسترسی به عناصر آرایه بکار می روند. کلیدهای مورد استفاده در آرایه های انجمنی، ممکن است عدد صحیح یا رشته باشند.

There are two ways to create an associative array:

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

or:

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

The named keys can then be used in a script:

Example

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
?>
```

Peter is 35 years old.

مرتب سازی آرایه ها Sorting Arrays

• PHP توابع متعددی را برای مرتب سازی آرایه ها ارائه می دهد.

- `sort()` - sort arrays in ascending order
 - `rsort()` - sort arrays in descending order
 - `asort()` - sort associative arrays in ascending order, according to the value
 - `ksort()` - sort associative arrays in ascending order, according to the key
 - `arsort()` - sort associative arrays in descending order, according to the value
 - `krsort()` - sort associative arrays in descending order, according to the key
-

Example

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

BMW
Toyota
Volvo

Example

```
<?php  
$numbers = array(4, 6, 2, 22, 11);  
sort($numbers);  
?>
```

2
4
6
11
22

هرتب سازى نزولى

- Descending Order - `rsort()`

Example

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

Volvo
Toyota
BMW

22
11
6
4
2

Example

```
<?php  
$numbers = array(4, 6, 2, 22, 11);  
rsort($numbers);  
?>
```

متغیرهای فوق سراسری Superglobals

• این متغیرها بدون توجه به محدوده و بدون استفاده از کلمه کلیدی global در همه جای برنامه قابل دسترسی هستند. این متغیرها عبارتند از:

- `$GLOBALS`
- `$_SERVER`
- `$_REQUEST`
- `$_POST`
- `$_GET`
- `$_FILES`
- `$_ENV`
- `$_COOKIE`
- `$_SESSION`

متغیرهای فوق سراسری Superglobals

Variable name	Description
<code>\$_SERVER</code>	Data about the currently running server.
<code>\$_ENV</code>	Data about the client's environment.
<code>\$_GET</code>	Data sent to the server by a <code>get</code> request.
<code>\$_POST</code>	Data sent to the server by a <code>post</code> request.
<code>\$_COOKIE</code>	Data contained in cookies on the client's computer.
<code>\$GLOBALS</code>	Array containing all global variables.

Fig. 19.12 | Some useful superglobal arrays.

متغیر \$GLOBALS

- PHP تمامی متغیرهای سراسری را در آرایه `$GLOBALS[index]` ذخیره می کند که `index` نام متغیر را نگهداری می کند.

Example

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

function addition() {
    $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}

addition();
echo $z;
?>
```



Code 42.txt

100

In the example above, since z is a variable present within the `$GLOBALS` array, it is also accessible from outside the function!

\$_GET

- یکی از روش های ارسال داده های فرم به اسکریپت سرویس دهنده می باشد.
زمانی که یک فرم، داده های خود را با متد `get` ارسال نماید، آنگاه این داده در اسکریپت مقصد با آرایه `$_GET` قابل دسترسی خواهد بود.

PHP `$_GET` can also be used to collect form data after submitting an HTML form with `method="get"`.

`$_GET` can also collect data sent in the URL.

\$_POST

- زمانی که یک فرم، داده های خود را با متد post به مقصد ارسال نماید، آنگاه این داده ها در اسکریپت مقصد با `$_POST` قابل دسترس خواهند بود.

Example

```
<html>
<body>

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

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_POST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>

</body>
</html>
```

Example

مثال

```
<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>
```

The output could be something like this:

```
Welcome John
Your email address is john.doe@example.com
```

To display the submitted data you could simply echo all the variables. The "welcome.php" looks like this:

```
<html>
<body>

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

</body>
</html>
```

The same result could also be achieved using the HTTP GET method:

Example

```
<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>
```

and "welcome_get.php" looks like this:

```
<html>
<body>

Welcome <?php echo $_GET["name"]; ?><br>
Your email address is: <?php echo $_GET["email"]; ?>

</body>
</html>
```

Name & Family:

Favoirit Color:

Next>

Name & Family:

Favoirit Color:

Next>

M.Nemati



Code 44.txt



Code 45.txt

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Document</title>
6  </head>
7  <body>
8      <form action="result.php" method="get">
9          Name & Family:
10         <input type="text" name="firstname" placeholder="Enter first name...">
11         <br>
12         <br>
13         Favoirit Color:
14         <input type="color" name="favcolor" >
15         <br>
16         <br>
17         <input type="submit" value="Next">
18     </form>
19 </body>
20 </html>

```

```

1  <?php
2
3  $name=$_GET["firstname"];
4  $color=$_GET["favcolor"];
5
6  echo "<h1 style='color:". $color .";'>". $name .
7
8  <"/h1>";
9
10 ?>

```

اعتبارسنجی فرم Form Validation

- زمان پردازش فرم های PHP، به امنیت بیاندیشید! در این مبحث نحوه ی پردازش فرم های PHP بصورتی که حداقل موارد امنیتی حفظ شود، نشان داده خواهد شد. اعتبار سنجی مناسب اطلاعات فرم، بمنظور جلوگیری از حمله هکرها دارای اهمیت است.

مثال

```
<> index.html x page2.php
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6 </head>
7 <body>
8   <form action="page2.php" method="get">
9     First Name:
10    <input type="text" name="fname">
11    <br>
12    <br>
13    <input type="submit" value="register">
14  </form>
15 </body>
16 </html>
17
```

```
<> index.html page2.php x
1 <?php
2   $fn=$_GET['fname'];
3   echo $fn;
4 ?>
5
6
```

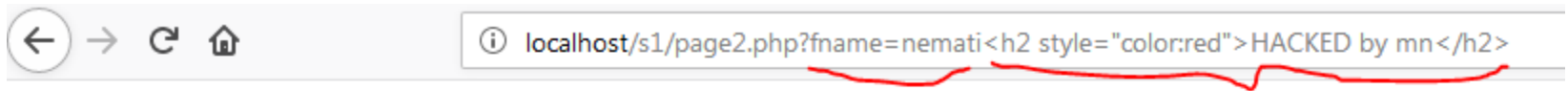
localhost/s1/index.html

First Name:

localhost/s1/page2.php?fname=nemati

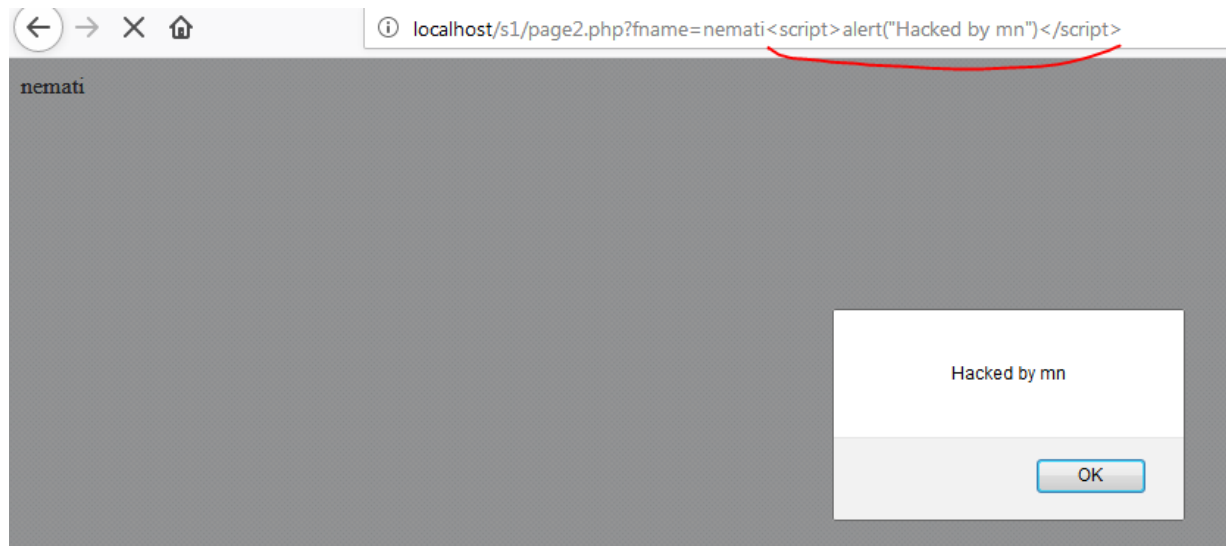
nemati

تغییر در داده های ارسالی



nemati

HACKED by mn



XSS

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in Web applications. XSS enables attackers to inject client-side script into Web pages viewed by other users.

Cross-site scripting یا XSS

XSS یک نوع قابلیت آسیب پذیری امنیت کامپیوتر است. معمولاً در برنامه های کاربردی web بکار می رود. XSS، هکرها را قادر می سازد تا صفحات وب را از طریق تزریق اسکریپت سمت client هک کنند.

دلیل به وجود آمدن این آسیب پذیری عدم اعتبارسنجی ورودی های کاربر می باشد، و مهاجم می تواند با تزریق اسکریپت های مخرب در سایت از این آسیب پذیری سوء استفاده کند.

تابع `htmlspecialchars()`

- با استفاده از تابع `htmlspecialchars()` می توان کاراکترهای خاص را به HTML entity تبدیل نمود. منظور این است که کاراکترهایی مثل علامت کوچکتر (`<`) و بزرگتر (`>`) در پارامتر ورودی را به `<` ; `>` تبدیل می کند. با این کار از حمله ی هکری که می خواهند از طریق تزریق HTML یا JavaScript اخلاص ایجاد کنند، جلوگیری می شود.

What is the `htmlspecialchars()` function?

The `htmlspecialchars()` function converts special characters to HTML entities. This means that it will replace HTML characters like `<` and `>` with `<` and `>`. This prevents attackers from exploiting the code by injecting HTML or Javascript code (Cross-site Scripting attacks) in forms.

نحوه حل مشکل

```
index.html  page2.php x
1  <?php
2      $fn=htmlspecialchars($_GET['fname']);
3      echo $fn;
4  ?>
5
6
```

localhost/s1/page2.php?fname=nemati<h2 style="color:red">HACKED by mn</h2>
nemati<h2 style="color:red">HACKED by mn</h2>

localhost/s1/page2.php?fname=nemati<script>alert("Hacked by mn")</script>
nemati<script>alert("Hacked by mn")</script>

نوشتن در قالب یک تابع

```
<> index.html  🐘 page2.php x
1  <?php
2      function test_input($data) {
3          $data = trim($data);
4          $data = stripslashes($data);
5          $data = htmlspecialchars($data);
6          return $data;
7      }
8
9      $fn = test_input($_GET['fname']);
10
11      echo $fn;
12  ?>
```

مثال کاملتر سایت w3schools

- https://www.w3schools.com/php/showphp.asp?filename=demo_form_validation_escapechar



Code 48.txt

PHP Form Validation Example

Name:

E-mail:

Website:

Comment:

Gender: ☐ Female ☐ Male ☐ Other

فیلدهای الزامی

- قصد داریم فرمی داشته باشیم که بعضی از فیلدهای آن الزامی باشد. در مثال زیر فیلدهای "نام"، "ایمیل" و "جنسیت" الزامی اند. این فیلدها نمی توانند خالی باشند و باید حتماً با یک مقدار تنظیم شوند.

Field	Validation Rules
Name	Required. + Must only contain letters and whitespace
E-mail	Required. + Must contain a valid email address (with @ and .)
Website	Optional. If present, it must contain a valid URL
Comment	Optional. Multi-line input field (textarea)
Gender	Required. Must select one

PHP Form Validation Example

Name:

E-mail:

Website:

Comment:

Gender: ☐ Female ☐ Male ☐ Other

فیلدهای الزامی

- در کد زیر، تعدادی متغیر جدید با نام های `$nameErr` و `$emailErr` و `$genderErr` و `$websiteErr` اضافه شده است. این متغیرهای خطا، متن خطای مورد نظر را برای فیلدهای الزامی در خود نگه می دارند. ما همچنین برای هر متغیر `POST_$` یک دستور `if ... else` اضافه کرده ایم. این دستور شرطی، چک می کند که اگر متغیر `POST_$` خالی است (با استفاده از تابع `(empty)`) یک متن خطای مناسب در متغیرهای خطا ذخیره می کند، و اگر خالی نبود، مقدار برگشتی تابع `(test_input)` در متغیر متناظرش ذخیره می شود.

در نظر گرفتن شرط برای فیلدهای الزامی



```
<?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"]);
    }
}
```


نمایش پیام های خطا

```
<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>
```



Code 49.txt

اعتبارسنجی فیلد ایمیل در PHP

- کد زیر، یک روش ساده برای چک کردن اینکه آیا فیلد "ایمیل" معتبر است یا نه را نشان می دهد. (باید شامل فرمت صحیح ایمیل باشد، همراه با علامت @ و .)

```
$email = test_input($_POST["email"]);  
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
    $emailErr = "Invalid email format";  
}
```

فیلتر ورودی ها در PHP

- تقریباً همه ی برنامه های تحت وب از محیط بیرون دریافت اطلاعات دارند که معمولاً این ورودی ها می تواند از طریق کاربران یا برنامه های دیگر مثل وب سرویس ها باشد، برای اطمینان از صحت اطلاعات ورودی، فیلترهای PHP کمک فراوانی به ما خواهند کرد.
- شما باید همیشه داده های ورودی را از فیلترها عبور دهید، در حقیقت فیلتر ورودی ها یکی از مهمترین مسائل امنیت برنامه هاست.

فیلتر ورودی ها در PHP

• ورودی های برنامه کدامند؟

- ✓ داده های دریافت شده از یک فرم HTML
- ✓ Cookieها
- ✓ داده های دریافت شده از وب سرویس ها
- ✓ متغیرهایی که روی سرور ایجاد می شوند مثل sessionها
- ✓ اطلاعاتی که از پایگاه داده بازیابی می شود.

انواع فیلتر

- به صورت کلی دو نوع فیلتر وجود دارد:
- فیلترهای اعتبار سنجی ($\text{Validating data} = \text{Determine if the data is in proper form}$)
 - ✓ برای معتبر کردن ورودی های کاربر استفاده می شوند.
 - ✓ برای قالب بندی قوانین محض استفاده می شود. (مثل معتبرسازی URL یا E-Mail)
 - ✓ اگر داده ورودی، اعتبار نداشته باشد FALSE وگرنه مقدار متغیر را بر می گرداند.
- فیلترهای اصولی ($\text{Sanitizing data} = \text{Remove any illegal character from the data}$)
 - ✓ وجود یا عدم وجود کاراکترهایی خاص در یک رشته را بررسی می کند.
 - ✓ برای قالب بندی داده ها نیست.
 - ✓ همیشه یک رشته برمی گرداند.

PHP filter_var() Function

- این تابع یک متغیر خاص را با یک صافی مخصوص فیلتر می کند. این تابع دو پارامتر را دریافت می کند، یکی متغیری که می خواهیم چک شود و دیگری نوع چک کردن آن متغیر است.

Sanitize a String

Example

```
<?php
$str = "<h1>Hello World!</h1>";
$newstr = filter_var($str, FILTER_SANITIZE_STRING);
echo $newstr;
?>
```

Sanitize and Validate an Email Address

Example

```
<?php
$email = "john.doe@example.com";

// Remove all illegal characters from email
$email = filter_var($email, FILTER_SANITIZE_EMAIL);

// Validate e-mail
if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
    echo("$email is a valid email address");
} else {
    echo("$email is not a valid email address");
}
?>
```



Code 50.txt

حفظ مقادیر فرم در PHP

- زمانی که کاربر، مقادیر فیلدهای فرم را تنظیم و روی دکمه submit کلیک می کند، تمام فیلدها، مقادیرشان را از دست می دهند. حالا تصور نمایید که کاربر، در ورود اطلاعات یک فیلد (مثلاً فیلد "ایمیل") خطایی داشته باشد، بنابراین باید از ابتدا شروع به وارد کردن اطلاعات کند...!
- بمنظور حفظ مقادیر فیلدهای فرم، برای فیلدهای متنی، باید یک کد PHP کوچک در خصوصیت value تگ input قرار دهیم.

حفظ مقادیر فرم در PHP

```
Name: <input type="text" name="name" value="<?php echo $name;?>">
<span class="error">* <?php echo $nameErr;?></span>
<br><br>
E-mail: <input type="text" name="email" value="<?php echo $email;?>">
<span class="error">* <?php echo $emailErr;?></span>
```



Code 51.txt

Sessions

- **SESSION** راهی است که از آن طریق می توان یکسری داده های خاص را زمانیکه از یک صفحه سایت به صفحه دیگری می رویم انتقال داده و به عبارتی به خاطر آورد. **SESSION** داده های مد نظر خود را **روی سرور** ذخیره می سازند.

شروع Session

- قبل از اینکه بتوانید اطلاعات کاربر، را در جلسه تان ذخیره کنید، ابتدا باید یک جلسه آغاز شود، تابع `session_start()` برای این منظور استفاده می شود.

Example

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

نکته

- تابع `session_start()` باید قبل از هر کد دیگری در صفحه قرار گیرد.

Note: The `session_start()` function must be the very first thing in your document. Before any HTML tags.

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>
```

از بین بردن SESSION

- برای از بین بردن جلسه ها در PHP می توان از توابع زیر استفاده کرد.

`session_unset()` ✓

`session_destroy()` ✓

To remove all global session variables and destroy the session, use `session_unset()` and `session_destroy()`:

```
<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();
?>
```