

PHP

Functions, Constants,
Arrays, include/require, String, Date,
Math functions, Redirection

how to create your own functions

- Functions can be written anywhere within a page.
- Function can be called anywhere within a page.

Syntax

```
function functionName(parameters)
{
    stmts;
}
```

```
<head>
<title>Function Example</title>
</head>
<body>
<?php
function myName ()
{
    print "<h1 style=\"color:#FF0000;font-family:Arial, Helvetica, sans-serif\">Ali</h1>";
}
function perimeter($radius){
    return 2* 3.1415*$radius;
}

myName(); // function call
$res = perimeter(4); //function call

echo "Perimeter: $res <br>";
echo "Perimeter: ".perimeter(4); //function call

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

Output



Ali

Perimeter: 25.132
Perimeter: 25.132



```
<?php
function myName() {
    print "<h1 style=\"color:#FF0000;font-family:Arial, Helvetica, sans-serif\">Ali</h1>";
}
function perimeter($radius) {
    return 2* 3.1415*$radius;
}
?>
```



Functions are at the top of the page

```
<html>
<head>
<title>Function Example</title>
</head>
<body>
<?php
myName();
$res = perimeter(4);
echo "Perimeter: $res <br>";
echo "Perimeter: ".perimeter(4);
?>
</body>
</html>
```

Ali

Perimeter: 25.132
Perimeter: 25.132

```
<html>
<head>
<title>Function Example</title>
</head>
<body>
<?php
myName();
$res = perimeter(4);
echo "Perimeter: $res <br>";
echo "Perimeter: ".perimeter(4);
?>
</body>
</html>
```



Functions are at the end of the page

```
<?php
function myName() {
    print "<h1 style=\"color:#FF0000;font-family:Arial, Helvetica, sans-serif\">Ali</h1>";
}
function perimeter($radius) {
    return 2* 3.1415*$radius;
}
?>
```

One function is at the top of the page



```
<?php
function myName () {
    print "<h1 style=\"color:#FF0000;font-family:Arial, Helvetica, sans-serif\">Ali</h1>";
}
?>
<html>
<head>
<title>Function Example</title>
</head>
<body>
<?php
myName(); // function call
$res = perimeter(4); //function call
echo "Perimeter: $res <br>";
echo "Perimeter: ".perimeter(4); //function call
?>
</body>
</html>
```

Ali

Perimeter: 25.132

Perimeter: 25.132

One function is at the bottom of the page



```
<?php
function perimeter($radius) {
    return 2* 3.1415*$radius;
}
?>
```

Passing Parameters to a function

```
<html>
<head>
<title>Function Example</title>
</head>
<body>
```

```
<?php
function add($val1, $val2){ // function can be placed anywhere within page
    return $val1 + $val2;
}
```

```
$var1 = 10;
$var2 = 20;
```

```
print "Total is ".add($var1, $var2); // calling function
```

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

Output



Pass by Reference

```
<html>
<head>
<title>Function Example</title>
</head>
<body>
<?php
$v1 = 10; //original variables
$v2 = 20;

print "Before $v1, $v2"; //10 20
swap($v1, $v2);
print "<br />After $v1, $v2"; //20 30

?>
</body>
</html>
<?php
function swap(&$val1, &$val2) {
    $tmp = $val1;
    $val1 = $val2;
    $val2 = $tmp;
}
?>
```

& is used to call a **parameter by reference**
Function swap(&\$val1, &\$val2)

Output



Before 10, 20
After 20, 10

Function and Variable Scope

- A variable defined in a function called **LOCAL variable, valid only in the function**
- A variable defined outside of a function is called **GLOBAL variable**, which is **accessible by any part of the php script.**

```

<html>
<?php
function test() {
    echo "Filename 2: $fn";
    $filename = $GLOBALS["fn"];
    echo "<br>Filename 3: $filename";
}
?>
<head>
<title>Variable Scope Example</title>
</head>
<body>
<?php
$fn = "c:/test";

echo "Filename 1: $fn <br>";
test();

?>
</body>
</html>

```

\$filename is
local variable



\$fn is **local** variable



To access **global** variable inside function



\$fn is **global** variable

Do not use \$ sign



`$GLOBALS["fn"]`

Output



Filename 1: c:/test
Filename 2:
Filename 3: c:/test

PHP Constant

```
<html>
<head>
<title>Constant Example</title>
</head>
<body>
```

```
<?php
```

Do not use \$ sign when define constant



```
define("PI", 3.1415);
define("MAX", 10000);
define("WELCOME_MSG", "Welcome to PHP");
```

```
//usage of constants
```

```
$radius = 3;
$x = 2 * PI * $radius;
echo $x;
echo "<br />";
echo WELCOME_MSG;
echo "<br /> ".MAX;
```

```
?>
```

```
</body>
```

```
</html>
```

Syntax

```
define("CONST_NAME", VALUE);
```

Output



18.849
Welcome to PHP
10000



Do not use \$ sign when access (use) constant

PHP Arrays

- An array is a special variable, which can store multiple values in one single variable.
- In PHP, there are three kind of arrays:
 - **Numeric array** - An array with a numeric index
 - **Associative array** - An array where each ID key is associated with a value
 - **Multidimensional array** - An array containing one or more arrays
- **Numeric Arrays**
 - A numeric array stores each array element with a numeric index.
 - There are two methods to create a numeric array.

```
<?php
```

```
$cars=array("BMW", "Hundai", "Opel", "Toyota", "Mercedes");
```

```
//or
```


```
$cars[0]="BMW";
```


```
$cars[1]="Hundai";
```

```
$cars[2]="Opel";
```

```
$cars[3]="Toyota";
```

```
$cars[4]="Mercedes";
```

 the index are automatically assigned (the index starts at 0):

 assign the index manually

```
echo $cars[0] . ", " . $cars[1] . ", " . $cars[2] . ", " . $cars[3] . ", " . $cars[4];  
?>
```

- **Associative Arrays**

- An associative array, each ID key is associated with a value.
- It looks like structure in C

```
<?php
$ages = array("Ali"=>25, "Zeynep"=>23, "Zafer"=>26);

echo $ages["Ali"].", ".$ages["Zeynep"].", ".$ages["Zafer"];

$customer = array("name"=>"Ali", "surname"=>"Gül", "account"=>12345);

echo "<br>Customer name: ".$customer["name"];

$customers = array();
$customers[0] = array("name"=>"Ali", "surname"=>"Gül", "account"=>12345);
$customers[1] = array("name"=>"Veli", "surname"=>"Gül", "account"=>67585);
$customers[2] = array("name"=>"Zeynep", "surname"=>"Can", "account"=>72512);

echo "<br />Customer name: ".$customers[0]["name"];
?>
```

25, 23, 26

Customer name: Ali

Customer name: Ali

Looping through PHP Elements

```
<?php
$colors = array();
$colors = array("Red", "Yellow", "Green");
//or
$colors[0] = "Red";
$colors[1] = "Yellow";
$colors[2] = "Green";
echo "COLORS: ";
for($i=0; $i<count($colors ); $i++){
    echo $colors [$i]." ";
}
//or
foreach ($colors as $color) {
    echo $color." ";
}
echo "<br>";

$ages = array(25,45,60,70);
echo "AGES: " ;
for($i=0; $i<count($ages); $i++){
    echo $ages[$i]." ";
}
//or
foreach ($ages as $age) {
    echo $age." ";
}
?>
```

count function returns
number of elements in an array

syntax

```
foreach ($array as $value)
{
    stmts;
}
```

<code>\$ages = array("Ali"=>25, "Zeynep"=>23, "Zafer"=>26);</code>	25, Ali, 25
	23, Zeynep, 23
	26, Zafer, 26
<code>foreach(\$ages as \$key=>\$val){</code>	
<code>echo \$ages[\$key].", ".\$key.", ".\$val."
;</code>	name: Ali
<code>}</code>	surname: Gül
	account: 12345
<code>\$customers[0] = array("name"=>"Ali", "surname"=>"Gül", "account"=>12345);</code>	
<code>\$customers[1] = array("name"=>"Veli", "surname"=>"Gül", "account"=>67585);</code>	name: Veli
<code>\$customers[2] = array("name"=>"Zeynep", "surname"=>"Can", "account"=>72512);</code>	surname: Gül
	account: 67585
<code>foreach(\$customers as \$customer){</code>	
<code>echo "
;</code>	name: Zeynep
<code>foreach(\$customer as \$key=>\$val){</code>	surname: Can
<code>echo "\$key: \$val
;</code>	account: 72512
<code>}</code>	
<code>}</code>	name: Ali
	surname: Gül
	account: 12345
<code>for(\$i=0; \$i<count(\$customers); \$i++){</code>	name: Veli
<code>echo "
;</code>	surname: Gül
<code>foreach(\$customers[\$i] as \$key=>\$val){</code>	account: 67585
<code>echo "\$key: \$val
;</code>	
<code>}</code>	name: Zeynep
<code>}</code>	surname: Can
	account: 72512

To display all global variables

```
foreach($GLOBALS as $key=>$val) {  
    echo "$key : $val <br />";  
}
```

```
GLOBALS : Array  
_ENV : Array  
HTTP_ENV_VARS : Array  
ALLUSERSPROFILE : C:\ProgramData  
APPDATA : C:\Windows\system32\config\systemprofile\AppData\Roaming  
CommonProgramFiles : C:\Program Files (x86)\Common Files  
CommonProgramFiles(x86) : C:\Program Files (x86)\Common Files  
CommonProgramW6432 : C:\Program Files\Common Files  
COMPUTERNAME : NESEOZPC  
ComSpec : C:\Windows\system32\cmd.exe  
FP_NO_HOST_CHECK : NO  
LOCALAPPDATA : C:\Windows\system32\config\systemprofile\AppData\Local  
NUMBER_OF_PROCESSORS : 4  
OS : Windows_NT
```

Changing, Adding, Removing

```
$colors = array("Red", "Yellow", "Green");
```

```
print_r($colors);
```

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

```
$color[1] = "Black"; //changing
```

```
print_r($colors);
```

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

```
Array ( [0] => Red [1] => Yellow [2] => Green )
```

```
Array ( [0] => Red [1] => Yellow [2] => Green )
```

```
Array ( [0] => Red [1] => Yellow [2] => Green [3] => White )
```

```
Array ( [0] => Red [1] => Yellow [2] => Green )
```

```
Array ( [0] => Red [1] => Yellow [2] => Green )
```

```
array_push($colors, "White"); //add to the end of the array
```

```
print_r($colors);
```

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

```
$v = array_pop($colors); // remove from end
```

```
print_r($colors);
```

Array sorting (sort, rsort)

```
$colors = array("Red","Yellow","Green");  
echo "1: ";  
print_r($colors);
```

```
sort($colors);  
echo "<br />2: ";  
print_r($colors);
```

```
sort($colors, SORT_STRING); // sort in ascending, SORT_STRING: compares item as String  
echo "<br />3: ";  
print_r($colors);
```

```
rsort($colors);  
echo "<br />4: ";  
print_r($colors);
```

```
rsort($colors, SORT_STRING); // sort in descending, SORT_STRING: compares item as String  
echo "<br />5: ";  
print_r($colors);
```

```
$ages = array(22,18,5);  
sort($ages, SORT_NUMERIC); // sort in ascending, SORT_NUMERIC: compares items as numeric  
echo "<br />6: ";  
print_r($ages);
```

```
rsort($ages, SORT_NUMERIC); // sort in descending, SORT_STRING: compares item as String  
echo "<br />7: ";  
print_r($ages);
```

```
1: Array ( [0] => Red [1] => Yellow [2] => Green )  
2: Array ( [0] => Green [1] => Red [2] => Yellow )  
3: Array ( [0] => Green [1] => Red [2] => Yellow )  
4: Array ( [0] => Yellow [1] => Red [2] => Green )  
5: Array ( [0] => Yellow [1] => Red [2] => Green )  
6: Array ( [0] => 5 [1] => 18 [2] => 22 )  
7: Array ( [0] => 22 [1] => 18 [2] => 5 )
```


Associative Array: Sorting (ksort, asort)

```
$furits = array("d"=>"Lemon", "a"=>"Orange", "b"=>"Banana", "c"=>"Apple");
```

```
echo "1: ";  
print_r($furits);
```

```
ksort($furits); // sort by key as ascending order  
echo "<br />2: ";  
print_r($furits);
```

```
krsort($furits); // sort by key as descending order  
echo "<br />3: ";  
print_r($furits);
```

- 1: Array ([d] => Lemon [a] => Orange [b] => Banana [c] => Apple)
- 2: Array ([a] => Orange [b] => Banana [c] => Apple [d] => Lemon)
- 3: Array ([d] => Lemon [c] => Apple [b] => Banana [a] => Orange)
- 4: Array ([c] => Apple [b] => Banana [d] => Lemon [a] => Orange)
- 5: Array ([a] => Orange [d] => Lemon [b] => Banana [c] => Apple)

```
asort($furits); // sort by value as ascending order  
echo "<br />4: ";  
print_r($furits);
```

```
arsort($furits); // sort by value as descending order  
echo "<br />5: ";  
print_r($furits);
```

Other Useful Array Functions

```
<?php
$colors = array("Red", "Yellow", "Green");
$key = array_search("Green", $colors); // search a value and returns the key
echo "1: $key <br />";

$key = array_search("Black", $colors); // search a value and returns the key
if($key == null){
    echo "2: NOT FOUND";
}
else{
    echo "2: FOUNDED";
}

$reversearray = array_reverse($colors);
echo "<br />3: ";
print_r($reversearray);

$other = array("Black", "Blue");
$resarray = array_merge($colors, $other);
echo "<br />4: ";
print_r($resarray);
```

1: 2
2: NOT FOUND
3: Array ([0] => Green [1] => Yellow [2] => Red)
4: Array ([0] => Red [1] => Yellow [2] => Green [3] => Black [4] => Blue)
5: FOUND
6: Array ([0] => Ali [1] => Zeynep [2] => Zafer)
7: Array ([0] => 25 [1] => 26 [2] => 23)
8: Array ([0] => Yellow [1] => Green [2] => Red)

```
if(in_array("Green", $colors)) //searches an array for a specific value
    echo "<br />5: FOUND";
else
    echo "<br />5: NOT FOUND";
```

```
$ages = array("Ali"=>25, "Zeynep"=>23, "Zafer"=>26);
$keys=array_keys($ages);
echo "<br />6: ";
print_r($keys);
```

```
shuffle($ages);
echo "<br />7: ";
print_r($ages);
```

```
shuffle($colors);
echo "<br />8: ";
print_r($colors);
?>
```

```
1: 2
2: NOT FOUND
3: Array ( [0] => Green [1] => Yellow [2] => Red )
4: Array ( [0] => Red [1] => Yellow [2] => Green [3] => Black [4] => Blue )
5: FOUND
6: Array ( [0] => Ali [1] => Zeynep [2] => Zafer )
7: Array ( [0] => 25 [1] => 26 [2] => 23 )
8: Array ( [0] => Yellow [1] => Green [2] => Red )
```

PHP INCLUDE/REQUIRE

- To break a project into modules
- Included file's content is copied.
- .php, .html, .txt files can be included
- Especially useful for header, footer, menu of pages
- Require is the same as include, but it exits if it can not find the file

mylib.php

```
<?php
$var1 = 15;
function add($v1, $v2){
    return $v1 + $v2;
}
?>
```

```
require "mylib.php";
```

use.php

```
<html>
<head>
<title>include example</title>
</head>
<body>
<?php
include "mylib.php";

$num1 = $var1 + 5;
echo add(10, $num1);
?>
</body>
</html>
```

Output

30

String

```
<?php
$text = "Hello World";

$tmp = strtolower($text);
echo "1: $tmp , $text";

$tmp = strtoupper($text);
echo "<br />2: $tmp , $text";

$text = "hello world";
$tmp = ucfirst($text);
echo "<br />3: $tmp , $text";

$text = "hello world";
$tmp = ucwords($text);
echo "<br />4: $tmp , $text";

$len = strlen($text);
echo "<br />5: length is $len";

$var1 = 10;
$var2 = 10.3;
printf("<br /> %d is %2.2f <br>", $var1, $var2);

?>
```

```
1: hello world , Hello World
2: HELLO WORLD , Hello World
3: Hello world , hello world
4: Hello World , hello world
5: length is 11
10 is 10.30
```

explode function

```
<?php
```

```
$text = "Hello World";
```

```
//explode function returns an array of string
```

```
//explode(delimiter, string)
```

```
$ar = explode(" ", $text);
```

```
print_r($ar);
```

```
Array ( [0] => Hello [1] => World )
```

```
1985-12-10
```

```
$birth = "10/12/1985";
```

```
Ali Korkmaz 139 Ankara
```

```
$dt = explode("/", $birth);
```

```
print "<br> ".$dt[2]."-".$dt[1]."-".$dt[0];
```

```
$data = "Ali:Korkmaz:139:Ankara";
```

```
list($name, $surname, $id, $city) = explode(":", $data);
```

```
echo "<br />$name $surname $id $city";
```

p, strcasecmp functions

```
$str1 = "Hello";  
$str2 = "Hello";  
  
if(strcmp($str1, $str2) == 0){  
    echo "<br />EQUALS";  
}  
elseif(strcmp($str1, $str2) > 0){  
    echo "<br />$str1 is greater than $str2";  
}  
else{  
    echo "<br />$str1 is smaller than $str2";  
}
```

```
$str1 = "Hello";  
$str2 = "Helxyz";  
  
if(strcasecmp($str1, $str2) == 0){  
    echo "<br />EQUALS";  
}  
elseif(strcasecmp($str1, $str2) > 0){  
    echo "<br />$str1 is greater than $str2";  
}  
else{  
    echo "<br />$str1 is smaller than $str2";  
}
```

```
$str1 = "Hello";  
$str2 = "HELLO";  
  
if(strncmp($str1, $str2, 1) == 0){  
    echo "<br />EQUALS";  
}  
elseif(strncmp($str1, $str2) > 0){  
    echo "<br />$str1 is greater than $str2";  
}  
else{  
    echo "<br />$str1 is smaller than $str2";  
}
```

EQUALS
EQUALS
EQUALS

trim, rtrim functions

```
$str = "\r ali \n";  
echo "<br>Value of str:$str."  
//trim function removes whitespaces (space, tab, newline, carriage return)  
$nstr = trim($str);  
echo "<br />Value of newstr:$nstr."  
  
$str = "\r ali \n";  
$nstr = rtrim($str, "\n"); // delete \n at the end  
echo "<br />Value of newstr:$nstr."
```

```
<html>  
<head>  
<title>String Example</title>  
</head>  
<body>  
<br>Value of str:  
    ali  
.<br />Value of newstr:ali.<br />Value of newstr:  
    ali    .</body>  
</html>
```

```
Value of str: ali .  
Value of newstr:ali.  
Value of newstr: ali .
```


Accessing a character

```
-  
<?php  
$str = "Hello World";  
  
echo $str[0].", ".$str[6];  
  
echo "<br />";  
  
echo $str{0}."", ".$str{6};  
  
$str[0] = "h";  
$str[6] = "w";  
  
echo "<br>$str";  
  
?>
```

H,W
H,W
hello world

Extracting string (substr)

```
<?php
$mystr = "There is a cat in the tree";
$sub = substr($mystr, 11); //
echo "1: $sub";

$sub = substr($mystr, 11, 3); //string, start index, #of chracter
echo "<br />2: $sub";

$sub = substr($mystr, -4); //start form the end
echo "<br />3: $sub";

$sub = substr($mystr, -4, 2);
echo "<br />4: $sub";

?>
```

1: cat in the tree
2: cat
3: tree
4: tr

Replacing (str_replace, preg_replace)

```
<?php
```

```
$mystr = "Hello World World! World";
```

```
$str = str_replace("World", "WORLD", $mystr);
```

```
//parameters: find, replace, string
```

```
echo "<br />1: $str, $mystr";
```

```
$str = preg_replace("/World/", "WORLD", $mystr, 2);
```

```
// 2 means first 2 occurrence will be replaced
```

```
echo "<br />2: $str, $mystr";
```

```
?>
```

1: Hello WORLD WORLD! WORLD, Hello World World! World
2: Hello WORLD WORLD! World, Hello World World! World

Date

- To get current date
 - `Date(string $format[, int timestamp])`

```
<?php
$today = date("d/m/y");
echo "1: $today";

$today = date("d-m-y");
echo "<br />2: $today";

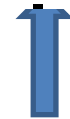
?>
```

1: 15/02/12 2: 15-02-12

Date Format: specifies how to return the result

- d - The day of the month (from 01 to 31)
- D - A textual representation of a day (three letters)
- j - The day of the month without leading zeros (1 to 31)
- l (lowercase 'l') - A full textual representation of a day
- N - The ISO-8601 numeric representation of a day (1 for Monday through 7 for Sunday)
- S - The English ordinal suffix for the day of the month (2 characters st, nd, rd or th. Works well with j)
- w - A numeric representation of the day (0 for Sunday through 6 for Saturday)
- z - The day of the year (from 0 through 365)
- W - The ISO-8601 week number of year (weeks starting on Monday)
- F - A full textual representation of a month (January through December)
- m - A numeric representation of a month (from 01 to 12)
- M - A short textual representation of a month (three letters)
- n - A numeric representation of a month, without leading zeros (1 to 12)
- t - The number of days in the given month
- L - Whether it's a leap year (1 if it is a leap year, 0 otherwise)
- o - The ISO-8601 year number
- Y - A four digit representation of a year
- y - A two digit representation of a year
- a - Lowercase am or pm
- A - Uppercase AM or PM
- B - Swatch Internet time (000 to 999)
- g - 12-hour format of an hour (1 to 12)
- G - 24-hour format of an hour (0 to 23)
- h - 12-hour format of an hour (01 to 12)
- H - 24-hour format of an hour (00 to 23)
- i - Minutes with leading zeros (00 to 59)
- s - Seconds, with leading zeros (00 to 59)
- e - The timezone identifier (Examples: UTC, Atlantic/Azores)
- I (capital i) - Whether the date is in daylight savings time (1 if Daylight Savings Time, 0 otherwise)
- O - Difference to Greenwich time (GMT) in hours (Example: +0100)
- T - Timezone setting of the PHP machine (Examples: EST, MDT)
- Z - Timezone offset in seconds. The offset west of UTC is negative, and the offset east of UTC is positive (-43200 to 43200)
- c - The ISO-8601 date (e.g. 2004-02-12T15:19:21+00:00)
- r - The RFC 2822 formatted date (e.g. Thu, 21 Dec 2000 16:01:07 +0200)
- U - The seconds since the Unix Epoch (January 1 1970 00:00:00 GMT)

```
$today = date("d/m/y");
```



format

Timestamp (mktime, time)

- Timestamp is a number of seconds from jan 1, 1970 at 00:00
- int **mktime**(hour, min, seconds, month, day, year): to create timestamp
- **time**() function to get curent timestamp

```
<?php
```

```
$mt = mktime(0,0,0,5,12,1987);
```

```
$gun = date("D",$mt);
```

```
echo "12.05.1987 is $gun";
```

```
$gun = date("l",$mt);
```

```
echo "<br />12.05.1987 is $gun";
```

```
//to get curent timestamp
```

```
$now = time();
```

```
echo "<br />Curent Timestamp Now: $now";
```

```
?>
```

12.05.1987 is Tue

12.05.1987 is Tuesday

Curent Timestamp Now: 1329327608

```
<?php
```

```
$n = 69.9235;
```

```
echo "Floor: ".floor($n);
```

```
echo "<br />Floor: ".floor(5.7);
```

```
echo "<br />Ceil: ".ceil($n);
```

```
echo "<br />Ceil: ".ceil(5.7);
```

```
echo "<br />Round: ".round($n);
```

```
echo "<br />Round: ".round(5.7);
```

```
echo "<br />Sqrt: ".sqrt($n);
```

```
echo "<br />Sqrt: ".sqrt(5.7);
```

```
echo "<br />abs: ".abs(-$n);
```

```
echo "<br />abs: ".abs(-5.7);
```

```
echo "<br />abs: ".pow($n, 3);
```

```
echo "<br />abs: ".pow(5.7, 3);
```

```
$ar = array(13,3,22,55,9);
```

```
echo "<br />Max: ".max($ar);
```

```
echo "<br />Min: ".min($ar);
```

```
echo "<br />Rand: ".rand(); //random int value between 0..32768
```

```
echo "<br />Rand: ".rand(10,50); // rand(minimumvalue, maximumvalue)
```

```
?>
```

Math Functions

Floor: 69

Floor: 5

Ceil: 70

Ceil: 5

Round: 70

Round: 6

Sqrt: 8.36202726616

Sqrt: 2.38746727726

abs: 69.9235

bs: 5.7

abs: 341876.678525

abs: 185.193

Max: 55

Min: 3

Rand: 16643

Rand: 33

Redirection (header)

- To send a page to another one,
 - Syntax: `header("Location:http://site.to.go/my.php");`

```
<html>
<head>
<title>Redirection Example</title>
</head>
<body>
<?php
    echo "<h1>Hello</h1>";
    header("Location:other.php");
    //Yoy will get warning message
?>
</body>
</html>
```



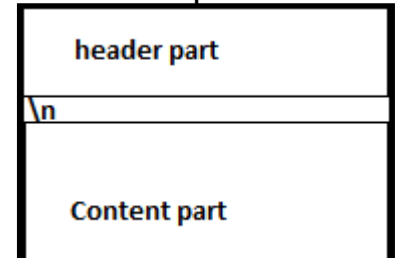
It creates a 200 OK packet



This results in error!!!

It does not work, because
header info must be sent prior to content

HTTP packet



Hello

Warning: Cannot modify header information - headers already sent by (output started at C:\AppServ\www\ctis256\week3\redirection\redirectionexwitherror.php:2) in C:\AppServ\www\ctis256\week3\redirection\redirectionexwitherror.php on line 3

- To solve redirection problem, we have to use “**output buffering**”.

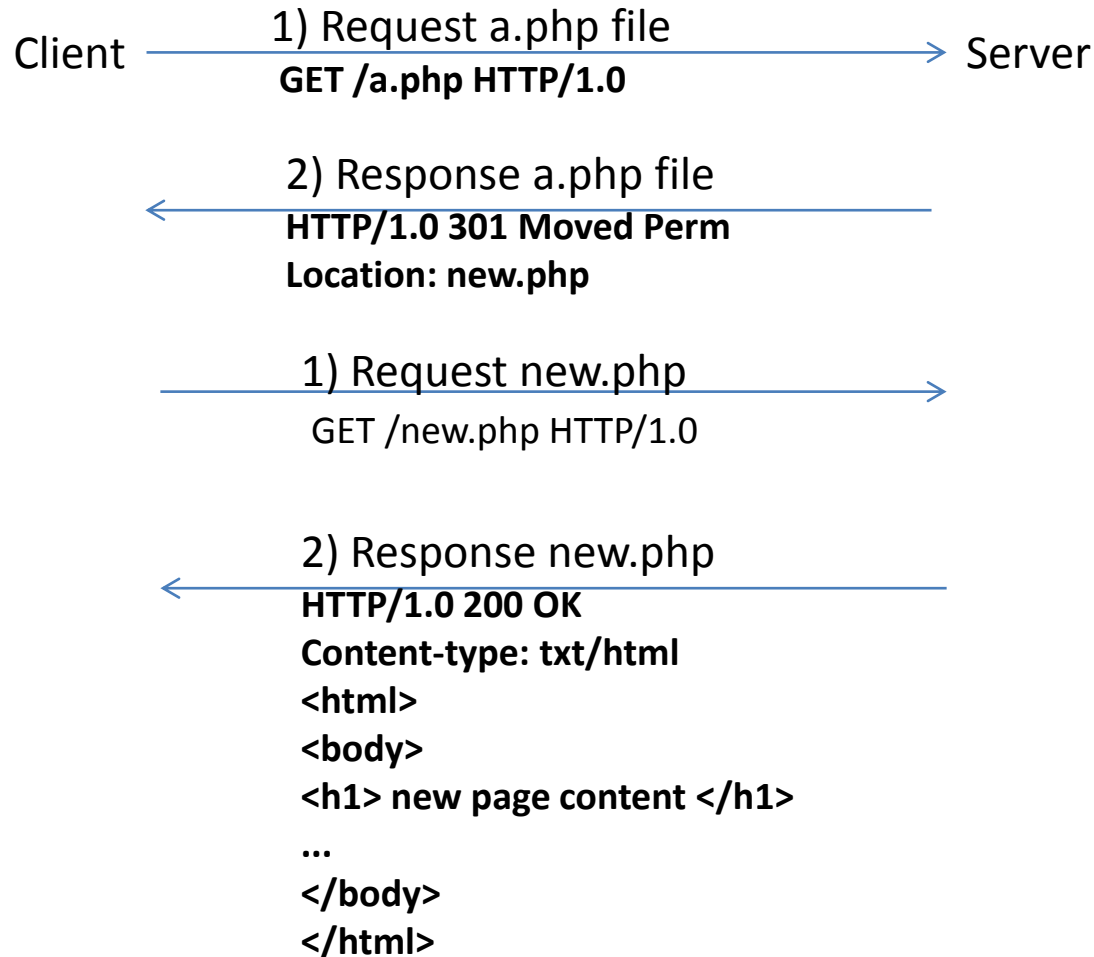
```
<?php
    ob_start();
?>
<html>
<head>
<title>Redirection Example</title>
</head>
<body>
<?php
    echo "<h1>Hello</h1>";
    header("Location:other.php");
?>
</body>
</html>
```

In short, in PHP, if redirection is used, **ob_start()** command **must be at the beginning of the php file**.

In this way, packet header does not send when content is displayed before header command.

With buffering, php reorders the HTTP packet in a way that header command will be in the packet

Data Flow with Redirection

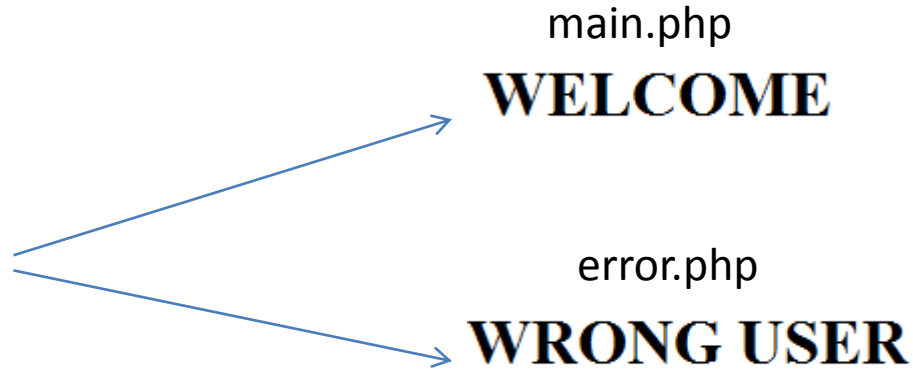


Excercise

Design following form
login.php

Login :

Password :



If login name is “**ali**” and password is “**ali123**”, then redirect to main.php ,
else
redirect to error.php or login.php with a parameter to print error in the form
(Hint: for error message use GET method)