PHP
Introduction

What is PHP?

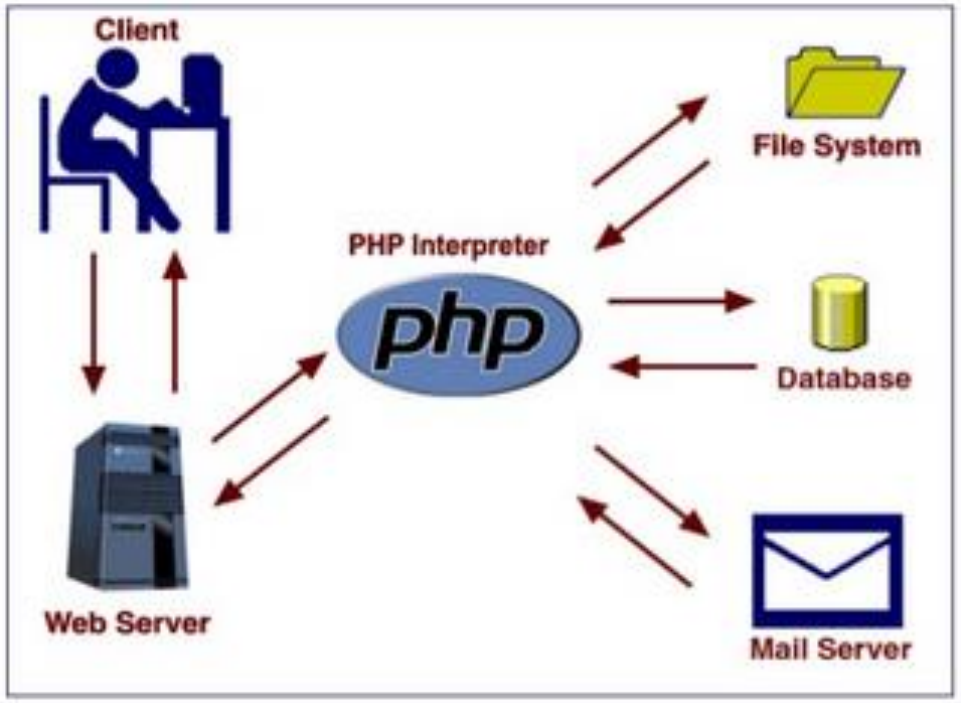
- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the serve

PHP File

- PHP files can contain text, HTML, CSS, JavaScript, and PHP code
- PHP code are executed on the server, and the result is returned to the browser as plain HTML
- PHP files have extension ".php"

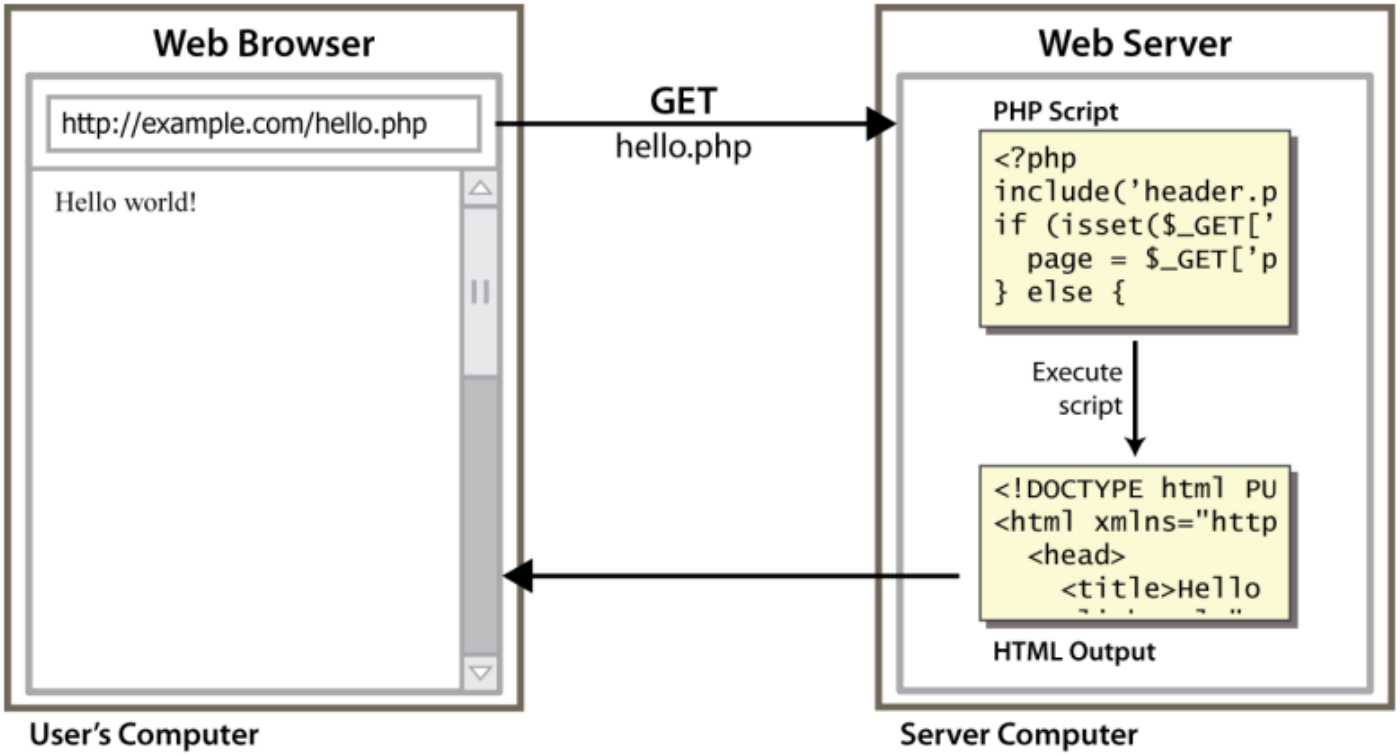
How PHP works

- when the user navigates in his browser to a page with a .php extension, the request is sent to a web server, which directs the request to the PHP interpreter.
- The PHP page is processed and the results is sent back to client.



Life cycle of a PHP web request

- The user requests a .php file.
- Server runs any script code inside.
- script produces output that becomes the response sent back to user's browser.



Using PHP we can:

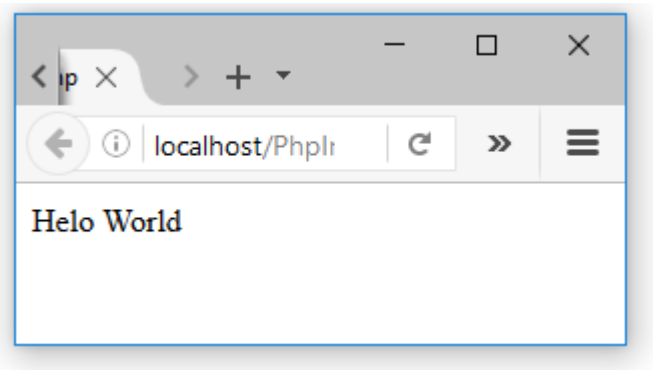
- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database

PHP Syntax

- A PHP script can be placed anywhere in the document.
- A PHP script starts with <?php and ends with ?>

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        echo "Hello World!";
        ?>
    </body>
</html>
```

- The PHP code is executed on the server, generating HTML which is sent back to the client.
- The hello world PHP page renders to an HTML page as bellow:



Comments in PHP

- A comment in PHP code is a line that is not read/executed as part of the program. Its only purpose is to be read by someone who is looking at the code.

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        // This is a single-line comment

        /*
          This is a multiple-lines comment block
          that spans over multiple
          lines
         */
        echo "Hello World!";
        ?>
    </body>
</html>
```

Declaring PHP Variables

- In PHP, a variable starts with the $ sign, followed by the name of the variable.

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $txt = "Hello world!";
        $x = 5;
        $y = 10.5;
        ?>
    </body>
</html>
```

Rules for PHP variables

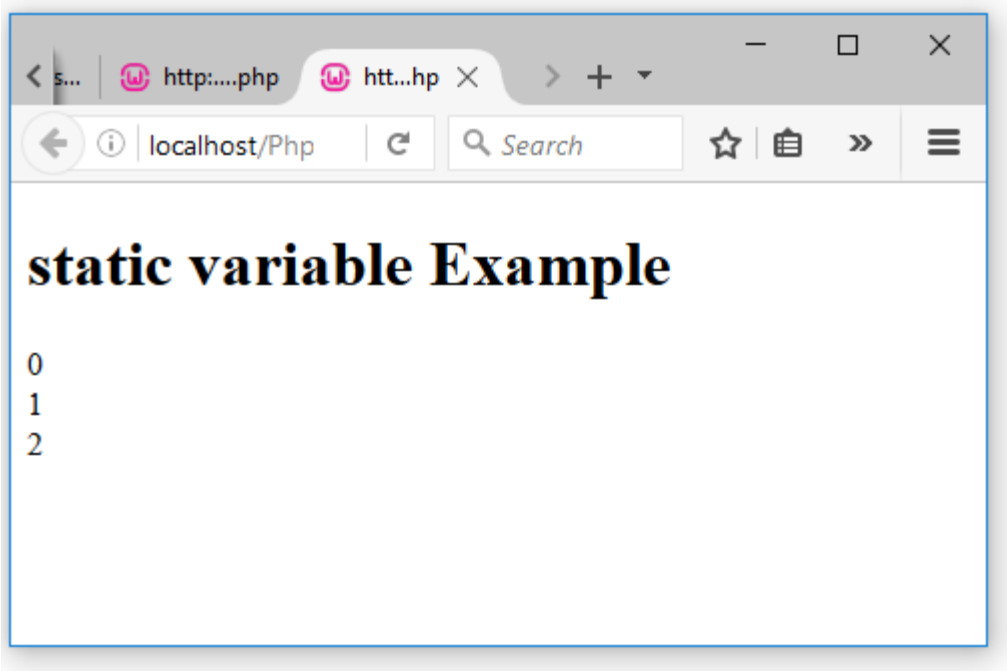
- A variable starts with the $ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive

Variables Scope

- PHP has three different variable scopes:
  - *local*
  - *global*
  - *Static*

    static variable is a local variable in a function that keeps its value after the function executed.

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>static variable Example</h1>
        <?php

        function myFunction() {
            static $x = 0;
            echo $x, "</br>";
            $x++;
        }

        myFunction();
        myFunction();
        myFunction();
        ?>
    </body>
</html>
```


http:....php
htt...hp
localhost/Php
Search
static variable Example
0
1
2

PHP Data Types

- PHP supports the following data types:
  - **Integers** – are whole numbers, without a decimal point, like 4195.
  - **Doubles** – are floating-point numbers, like 3.14159 or 49.1.
  - **Booleans** – have only two possible values either true or false.
  - **NULL** – is a special type that only has one value: NULL.
  - **Strings** – are sequences of characters, like 'PHP supports string operations.'
  - **Arrays** – are named and indexed collections of other values.
  - **Objects** – are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class.
  - **Resources** – are special variables that hold references to resources external to PHP (such as database connections).

Object Example

```
<?php
class foo
}
     function do_foo()
}
       echo "Doing foo.";
{
{

$bar = new foo;
$bar->do_foo();
?
```

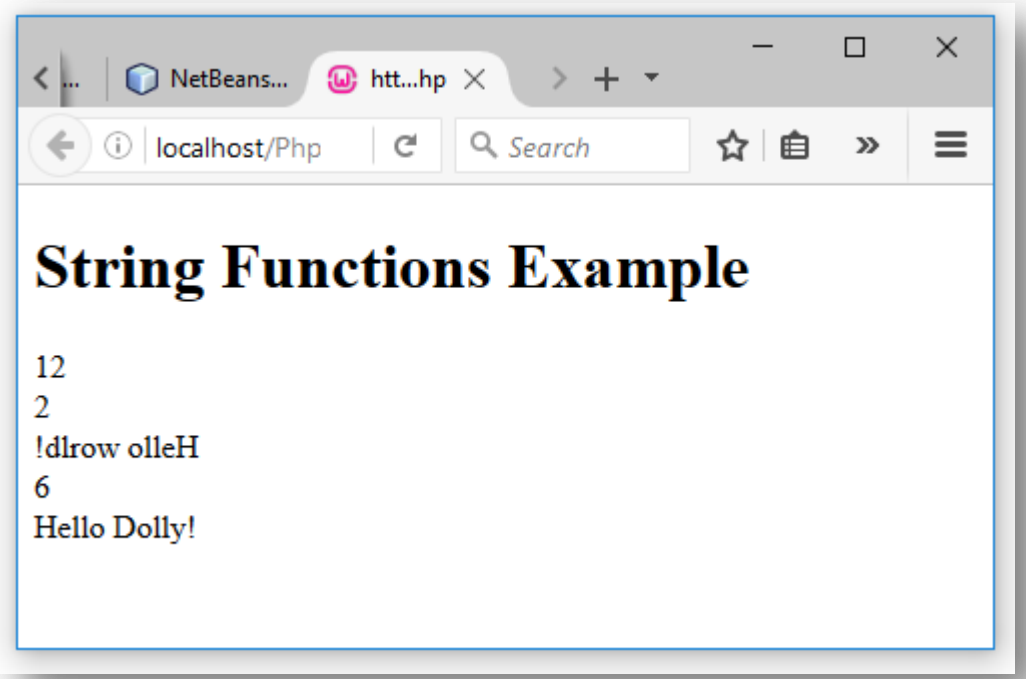
Resource Example

```
<?php
$c = mysql_connect();

$fp = fopen("foo", "w");

<?
```

PHP String Functions

Function	Uses	Example
strlen()	returns the length of a string.	echo *strlen*("Hello world!"); // outputs 12
str_word_count()	counts the number of words in a string	echo *str_word_count*("Hello world!"); // outputs 2
strrev()	reverses a string	echo *strrev*("Hello world!"); // outputs !dlrow olleH
strpos()	searches for a specific text within a string	echo *strpos*("Hello world!", "world"); // outputs 6
str_replace()	replaces some characters with some other characters in a string.	echo *str_replace*("world", "Dolly", "Hello world!"); // outputs Hello Dolly!

```
<!DOCTYPE html>
<!--
string_functions.php
-->
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>String Functions Example</h1>
        <?php
        echo strlen("Hello world!"),"</br>";
        echo str_word_count("Hello world!"),"</br>";
        echo strrev("Hello world!"),"</br>"; //
        echo strpos("Hello world!", "world"),"</br>";
        echo str_replace("world", "Dolly", "Hello world!");
        ?>
    </body>
</html>
```


NetBeans...
htt...hp
localhost/Php
Search
String Functions Example
12
2
!dlrow olleH
6
Hello Dolly!

PHP Constants

- A constant is an identifier for a value and its value cannot be changed during the script.
- The define() function is used to create a constant.

define(name, value, case-insensitive)

define(“PI", “3.14");

- Constants are automatically global and can be used across the entire script.

```php
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        define("PI", 3.14);

        function circleArea() {
            $area = PI * 10 * 10;
            echo $area;
        }

        circleArea();
        ?>
    </body>
</html>
```

PHP Operators

- Operators are used to perform operations on variables and values.
- PHP divides the operators in the following groups:
  - Arithmetic operators
  - Assignment operators
  - Comparison operators
  - Increment/Decrement operators
  - Logical operators
  - String operators
  - Array operators

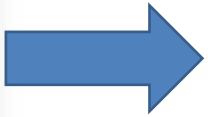
Arithmetic operators

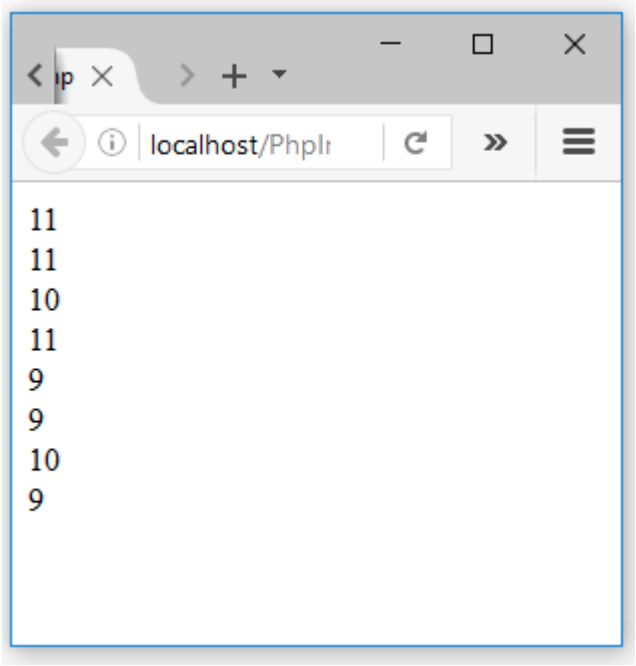
Operator	Description	Example
+	Addition	$j + 1
-	Subtraction	$j - 6
*	Multiplication	$j * 11
/	Division	$j / 4
%	Modulus (division remainder)	$j % 9
++	Increment	++$j
--	Decrement	--$j

Assignment operators

Operator	Example	Equivalent to
=	$j = 15	$j = 15
+=	$j += 5	$j = $j + 5
-=	$j -= 3	$j = $j - 3
*=	$j *= 8	$j = $j * 8
/=	$j /= 16	$j = $j / 16
.=	$j .= $k	$j = $j . $k
%=	$j %= 4	$j = $j % 4

```php
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $x = 10;
        echo ++$x, "<br/>";
        echo $x, "<br/>";

        $y = 10;
        echo $y++, "<br/>";
        echo $y, "<br/>";

        $z = 10;
        echo --$z, "<br/>";
        echo $z, "<br/>";

        $w = 10;
        echo $w--, "<br/>";
        echo $w, "<br/>";
        ?>
    </body>
</html>
```





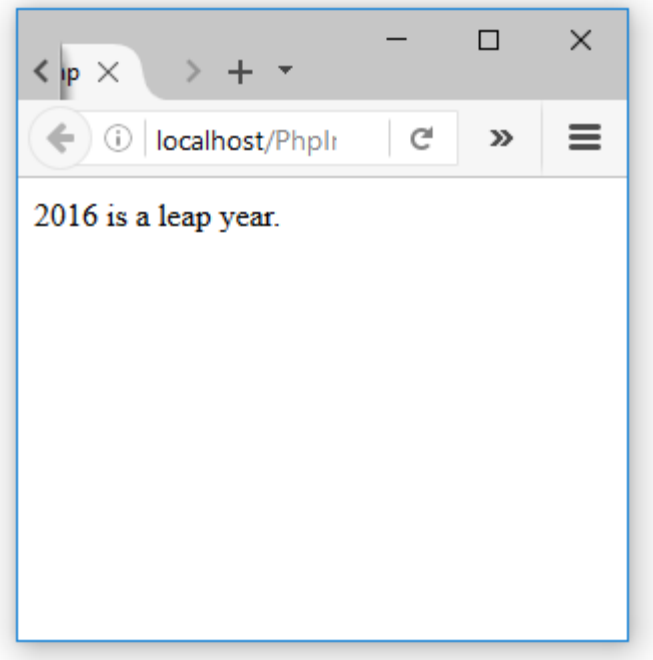
Comparison operators

Operator	Description	Example
==	Is *equal* to	`$j == 4`
!=	Is *not equal* to	`$j != 21`
>	Is *greater than*	`$j > 3`
<	Is *less than*	`$j < 100`
>=	Is *greater than or equal* to	`$j >= 15`
<=	Is *less than or equal* to	`$j <= 8`

Logical operators

Operator	Description	Example
&&	*And*	`$j == 3 && $k == 2`
and	Low-precedence *and*	`$j == 3 and $k == 2`
||	*Or*	`$j < 5 || $j > 10`
or	Low-precedence *or*	`$j < 5 or $j > 10`
!	*Not*	`! ($j == $k)`
xor	*Exclusive or*	`$j xor $k`

```php
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $year = 2016;
        // Leap years are divisible by 400 or by 4 but not 100
        if (($year % 400 == 0) || (($year % 100 != 0) && ($year % 4 == 0))) {
            echo "$year is a leap year.";
        } else {
            echo "$year is not a leap year.";
        }
        ?>
    </body>
</html>
```


localhost/Phpl
2016 is a leap year.

The if Statement

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $bank_balance = 5000;
        if ($bank_balance >= 500) {
            $zakat = ($bank_balance / 1000) * 25;
            echo $zakat;
        }
        ?>
    </body>
</html>
```

The if else Statement

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $bank_balance = 5000;
        if ($bank_balance >= 500) {
            $zakat = ($bank_balance / 1000) * 25;
            echo $zakat;
        } else {
            echo "No Zakat";
        }
        ?>
    </body>
</html>
```

The elseif Statement

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $x = 15;
        $y = 15;
        if ($x > $y) {
            echo "X is greather than y";
        } elseif ($y > $x) {

            echo "X is less than y";
        } else {
            echo "X is equal to y";
        }
        ?>
    </body>
</html>
```

The switch Statement

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $page ="News";
        switch ($page) {
            case "Home":
                echo "You selected Home";
                break;
            case "About":
                echo "You selected About";
                break;
            case "News":
                echo "You selected News";
                break;
            case "Login":
                echo "You selected Login";
                break;
            case "Links":
                echo "You selected Links";
                break;
        }
        ?>
    </body>
</html>
```

The ? Operator

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $money =4000;
        echo $money >= 5000 ? "Pay Zakat" : "No zakat";
        ?>
    </body>
</html>
```

while Loops

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>while Example</h1>
        <?php
        $x = 1;
        while ($x <= 10) {
            echo $x;
            $x = $x + 1;
        }
        ?>
    </body>
</html>
```

do...while Loops

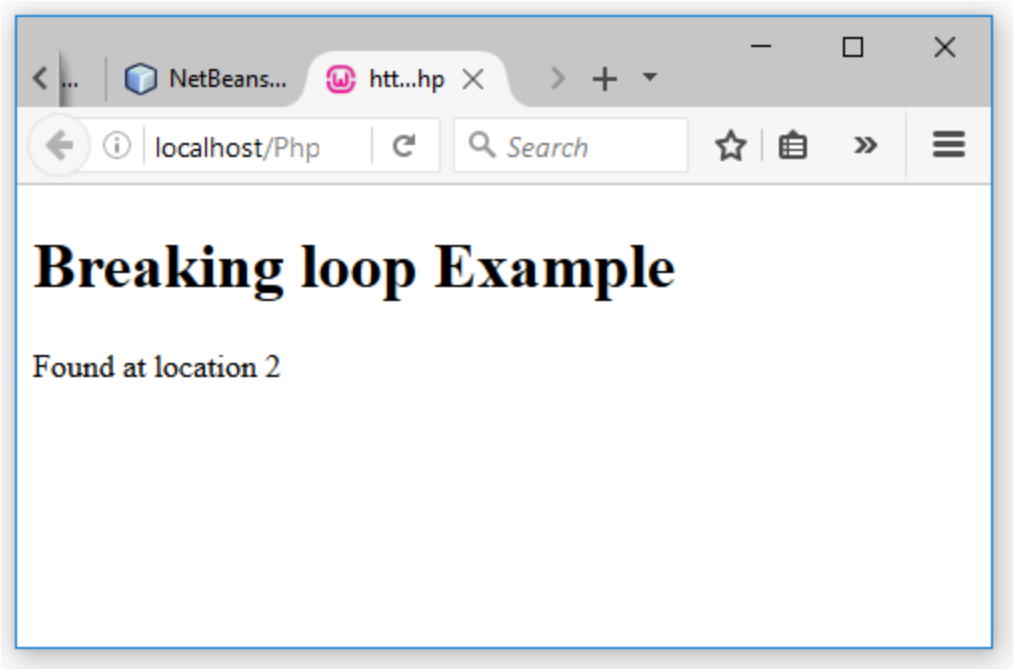
```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>do..while Example</h1>
        <?php
        $x = 1;
        do {
            echo $x;
            $x = $x + 1;
        } while ($x <= 10);
        ?>
    </body>
</html>
```

for Loops

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>For loop Example</h1>
        <?php
        for ($x = 0; $x <= 10; $x++) {
            echo $x;
        }
        ?>
    </body>
</html>
```

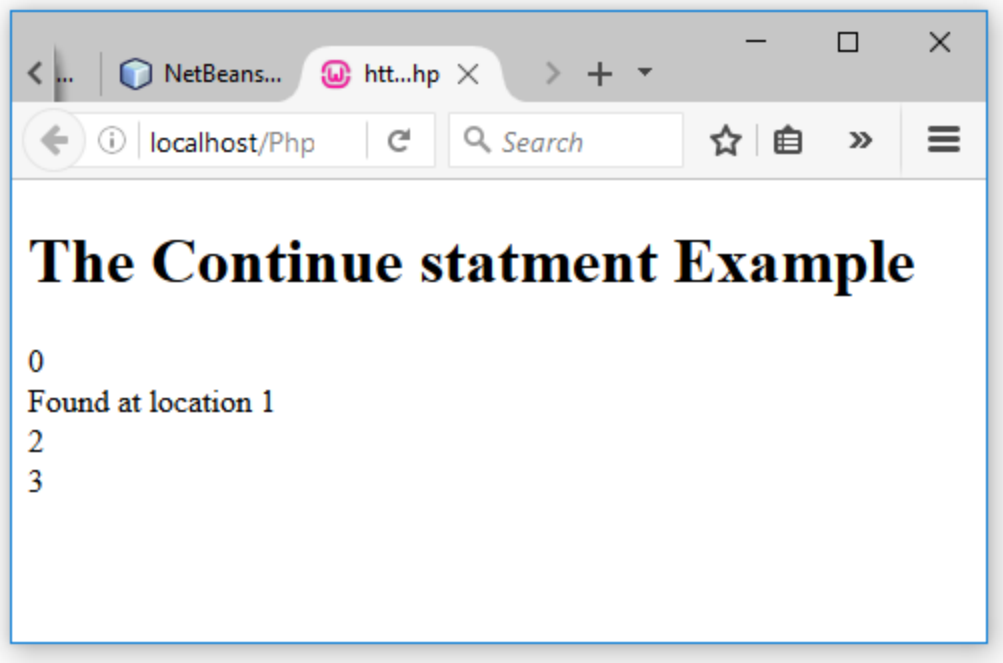
Breaking Out of a Loop

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>Breaking loop Example</h1>
        <?php
        $colors = array("red", "green", "blue", "yellow");
        for ($i = 0;$i <= 4;   $i++) {
            if ($colors[$i] == "blue") {
                echo"Found at location " , $i;
                break;
            }
        }
        ?>
    </body>
</html>
```


NetBeans...
htt...hp
localhost/Php
Search
Breaking loop Example
Found at location 2

The continue Statement

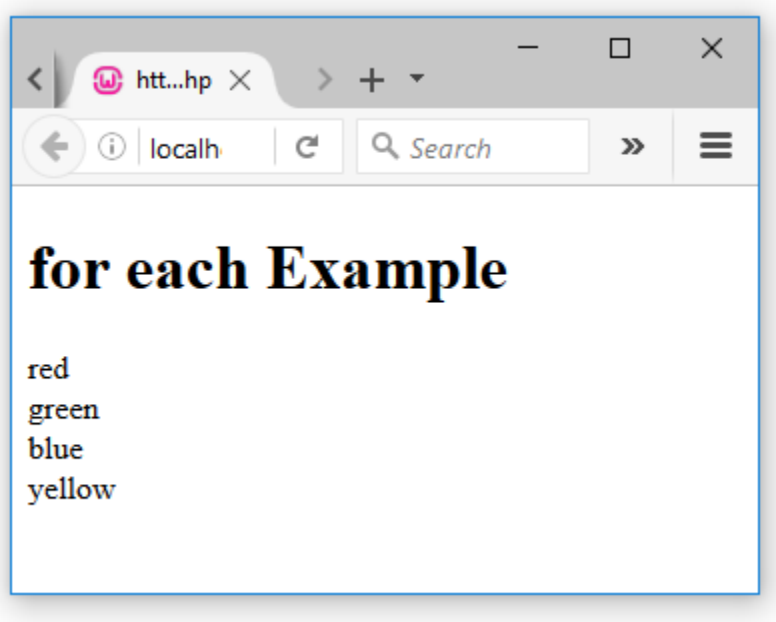
```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>The Continue statment Example</h1>
        <?php
       $colors = array("red", "green", "blue", "yellow");
        for ($i = 0;$i < 4;   $i++) {
            if ($colors[$i] == 'green') {
                echo "Found at location " , $i,"</br>";
                continue;
            }
            echo $i,"</br>";
        }
        ?>
    </body>
</html>
```


NetBeans...
htt...hp
localhost/Php
Search
The Continue statment Example
0
Found at location 1
2
3

foreach Loop

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

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $colors = array("red", "green", "blue", "yellow");

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


htt...hp
localh
Search
for each Example
red
green
blue
yellow

Thanks!