Introduction to PHP (Server Side Script)

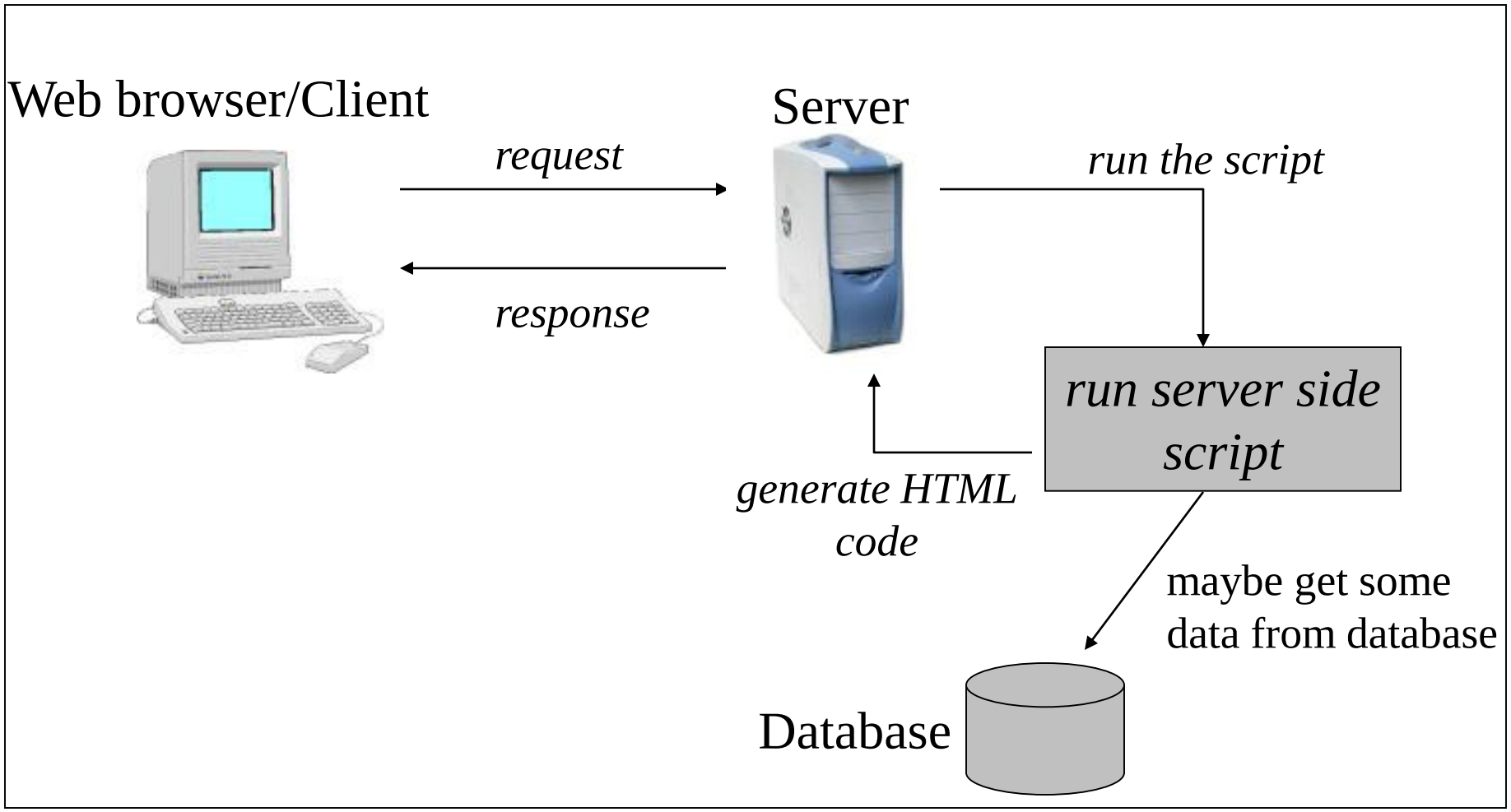
Outline

- ✖Introduction to Server Side Script
- ✖Introduction to PHP
- ✖Variables and Data Types
- ✖Operators
- ✖Conditional Statements and Loops
- ✖Functions
- ✖Function Argument
- ✖Array
- ✖FOREACH Loop

Introduction to Server Side Script

- Script/code that executed on the server to generate responses for clients.
- Server side script capabilities:
  - allows creation of interactive dynamic websites
  - modifies HTML code on the server before sent to client
  - access databases
  - responds to user input.
- Lots of examples: PHP, ASP, ASP.Net, JSP, CGI

How Server Side Script Works



Introduction to PHP

- PHP : recursive acronym for "***P****HP* ***H****ypertext* ***P****reprocessor*“
- Developed in 1995
- Widely-used Open Source scripting language that enables to create interactive web sites and can be embedded into HTML.
- PHP provides many great features:
  - free, easy to learn, fun to use, connects to most databases
  - provides many levels of security
  - available on most systems: Windows, Linux, Mac OS, etc
  - lots of built-in functions

Getting PHP to Work

- Three important components need to be installed on the computer system:
  - ✓ **Web server** – e.g. ***Apache*** (most widely used → free), IIS, PWS
  - ✓ **PHP software distribution** (parser)
    - Latest stable release is ***PHP* 5.5.19** and **5.6.3**
  - ✓ **Database** – e.g. ***MySQL*** (most popular RDBMS used over the Internet → free, fast, flawless), Oracle, Sybase

PHP Basic Syntax

- All PHP code must be included inside one of these three special markup tags:
- PHP files must be saved with a php extension, e.g.: .php, .php3 or .php4
- Every statement in PHP must be terminated with a semicolon (;)
- PHP is not generally case-sensitive
- Characters to indicate Comments:
  - // or # : single-line comment
  - /* *Here is an introduction to server side script code*
    inside a multi-line comment */ : multi-line comment

1. <?PHP code goes here ?>

```
<?php        PHP code goes here      ?>
<script language="php"> PHP code goes here   </script>
```

Example 1

- Save as exe1.php

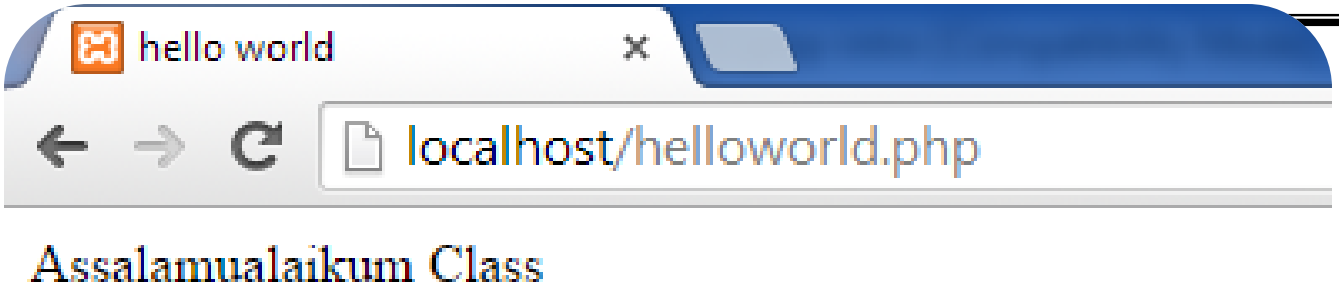
```
<html>
<head>

<title>hello world</title>
</head>

<body>
<p><?php echo “Assalamualaikum Class";
?></p>
</body>

</html>
```

Output 1


hello world
localhost/helloworld.php
Assalamualaikum Class

Variables and Data Types

- A variable contains a piece of data.
  - e.g.: number of records in a database, person's name,
- PHP is a "loosely typed" language.
  - a single variable may contain any type of data.
  - e.g.: a number (integer/floating point),
    a string of text or Boolean values.
  - not need to declare a variable to be a specific data type
- All variables in PHP begin with a $ (dollar sign).
- Variable names are case sensitive.

```
$name = "Ahmad";      // assign a string "Ahmad" to $name
$x = 45;              // assign an integer 45 to $x
```

VARIABLES IN PHP

PHP is a Loosely Typed Language

- In PHP, a variable does not need to be declared before adding a value to it.
- In the example above, you see that you do not have to tell PHP which data type the variable is.
- PHP automatically converts the variable to the correct data type, depending on its value.
- In a strongly typed programming language, you have to declare (define) the type and name of the variable before using it.
- In PHP, the variable is declared automatically when you use it.

VARIABLES IN PHP

Naming Rules for Variables

- A variable name must start with a letter or an underscore "_"
- A variable name can only contain alpha-numeric characters and underscores (a-z, A-Z, 0-9, and _)
- A variable name should not contain spaces. If a variable name is more than one word, it should be separated with an underscore (`$my_string`), or with capitalization (`$myString`)

Operators

- PHP supports several different types of operators that can be used in simple or complex expressions.

❑ Arithmetic Operators

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus

Example 2

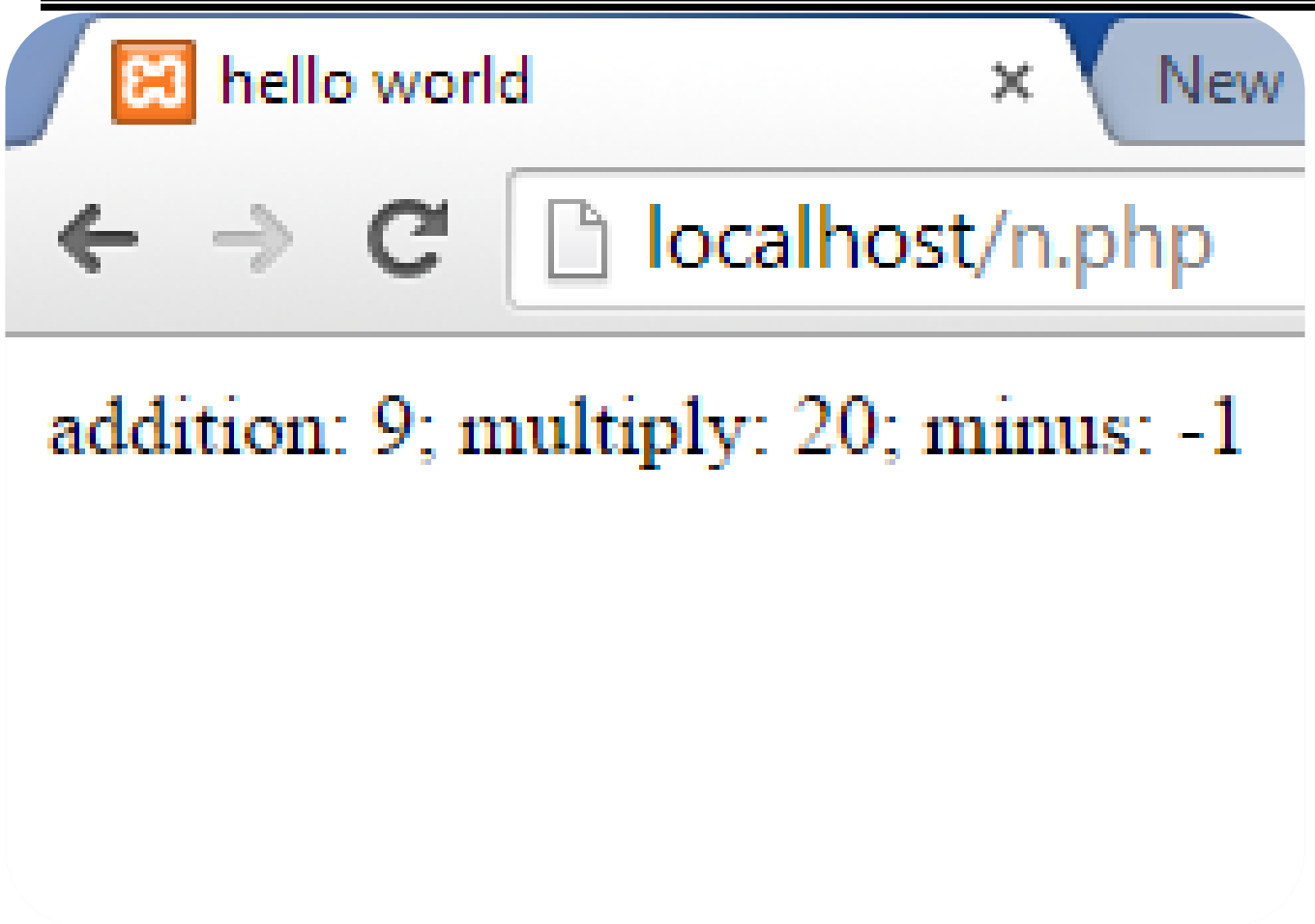
- Save as exe2.php

```
<?php
$x = 4;
$y = 5;

// Arithmetic operation
   $a = $x + $y;
   $b = $x * $y;
   $c = $x - $y;

echo "addition: $a;  multiply: $b;  minus: $c";
?>
```

Output 2


hello world
New
localhost/n.php
addition: 9; multiply: 20; minus: -1

Operators

- Comparison Operators

Operator	Comparative Test
==	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

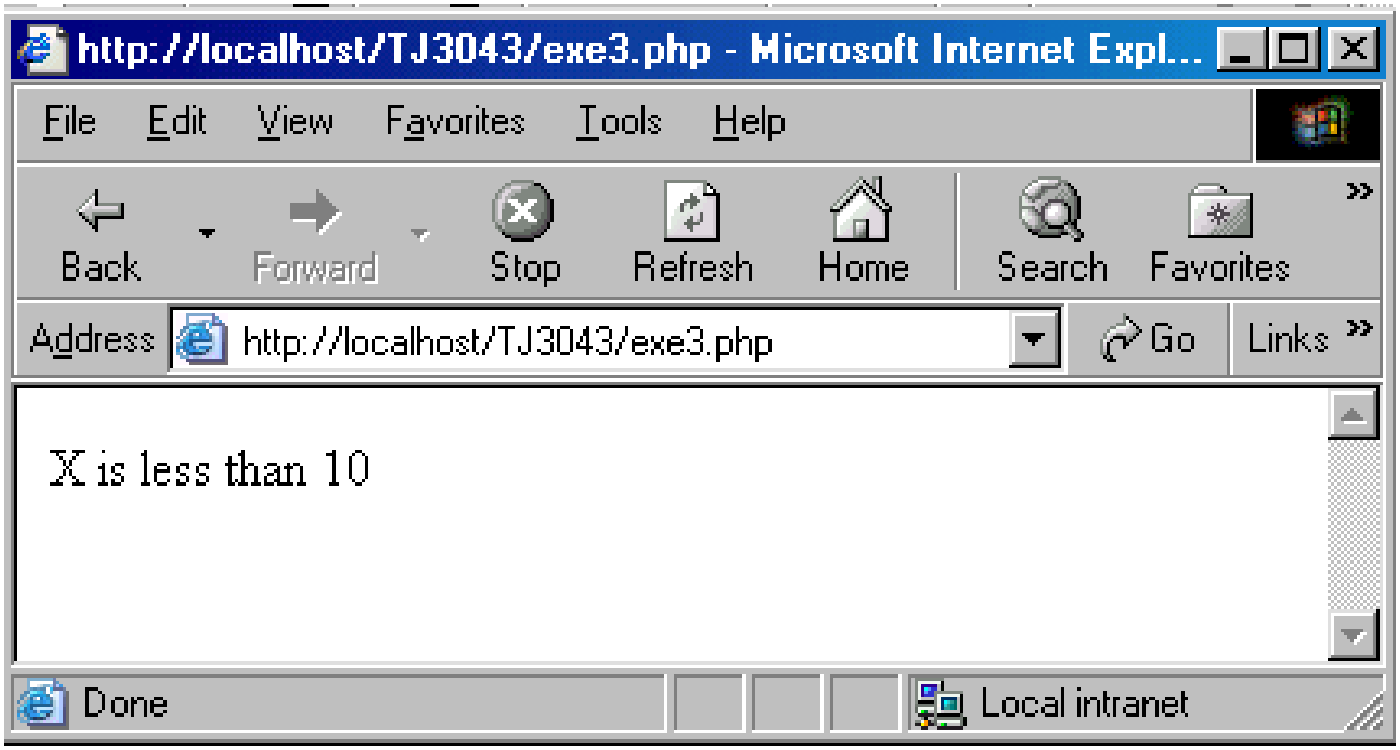
Example 3

- Save as exe3.php

```
<?php
$x = 8;

if ($x < 10)
        echo "X is less than 10";
else
        echo "X is greater than 10";
?>
```

Output 3


http://localhost/TJ3043/exe3.php - Microsoft Internet Expl...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe3.php
Go Links
X is less than 10
Done
Local intranet

Operators

- Logical Operators

Operator	Operation
&&	And
and	And
||	Or
or	Or
!	Not
Xor	Exclusive Or

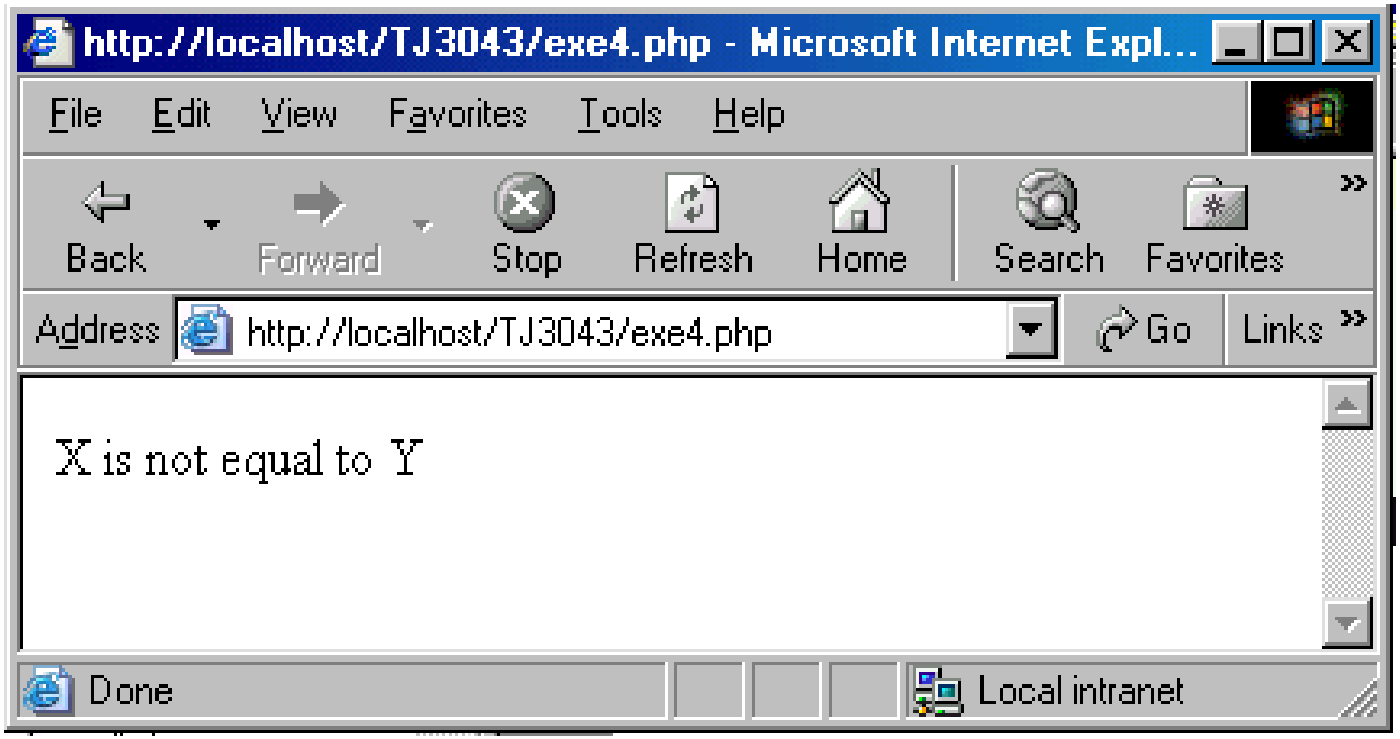
Example 4

- Save as exe4.php

```
<?php
$x = 4;
$y = 5;

if ($x == 4  &&  $y == 4)
      echo "X is equal to Y";
else
      echo "X is not equal to Y";
?>
```

Output 4


http://localhost/TJ3043/exe4.php - Microsoft Internet Expl...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe4.php
Go Links
X is not equal to Y
Done
Local intranet

Conditional Statements and Loops

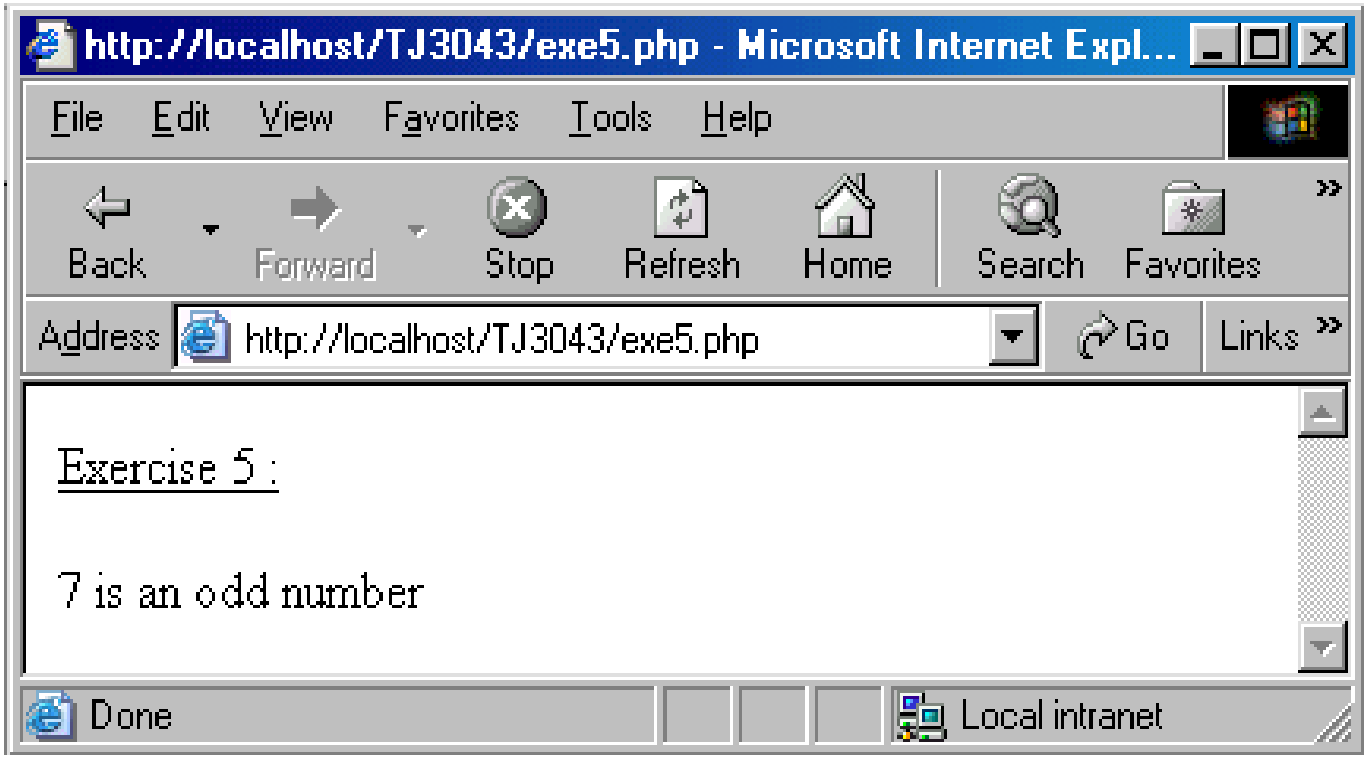
- Conditional *if* statement
  - The statement following the evaluation will only be executed when the expression is true.
  - The code to be executed may contain multiple statements if they are enclosed in a pair of curly brackets to form a **statement block**.

- Save as exe5.php

```
<?php
$num = 7;

if ($num % 2 != 0)
{   echo “<u> Exercise 5 : </u>";
    echo "<p>";
    echo "$num is an odd number";    }
?>
```

Output 5


http://localhost/TJ3043/exe5.php - Microsoft Internet Expl...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe5.php
Go Links
Exercise 5 :
7 is an odd number
Done
Local intranet

Conditional Statements and Loops

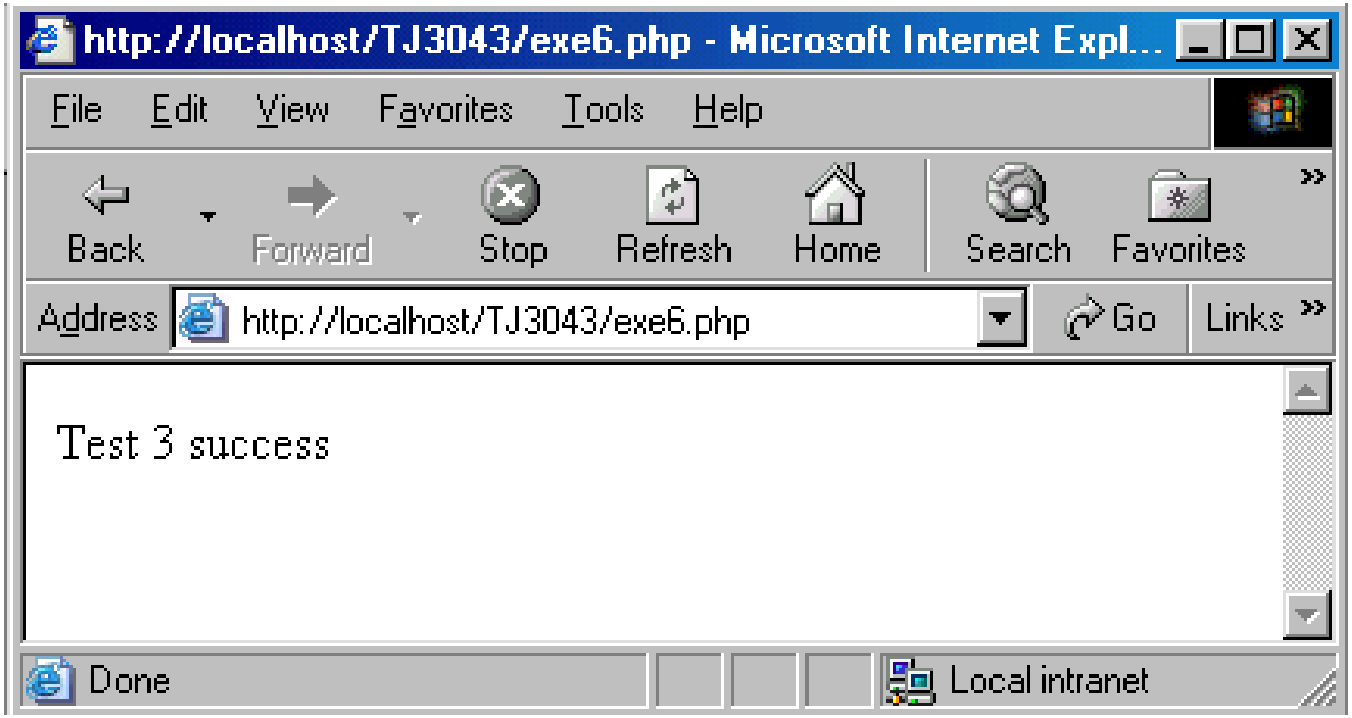
- Conditional *if-else and elseif* statement
  - ***else*** is used with an ***if*** statement to provide alternative code to execute in the event that the tested expression is found to be ***false***.

- Save as exe6.php

```
<?php
$num = 2;       $bool = false;

if ($num == 1 and $bool == true)
            echo "Test 1 success";
elseif ($num == 2 and $bool == true)
            echo "Test 2 success";
elseif ($num == 2 and $bool == false)
            echo "Test 3 success";
else
            echo "Test 4 success";     ?>
```

Output 6


http://localhost/TJ3043/exe6.php - Microsoft Internet Expl...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe6.php
Go Links
Test 3 success
Done
Local intranet

Conditional Statements and Loops

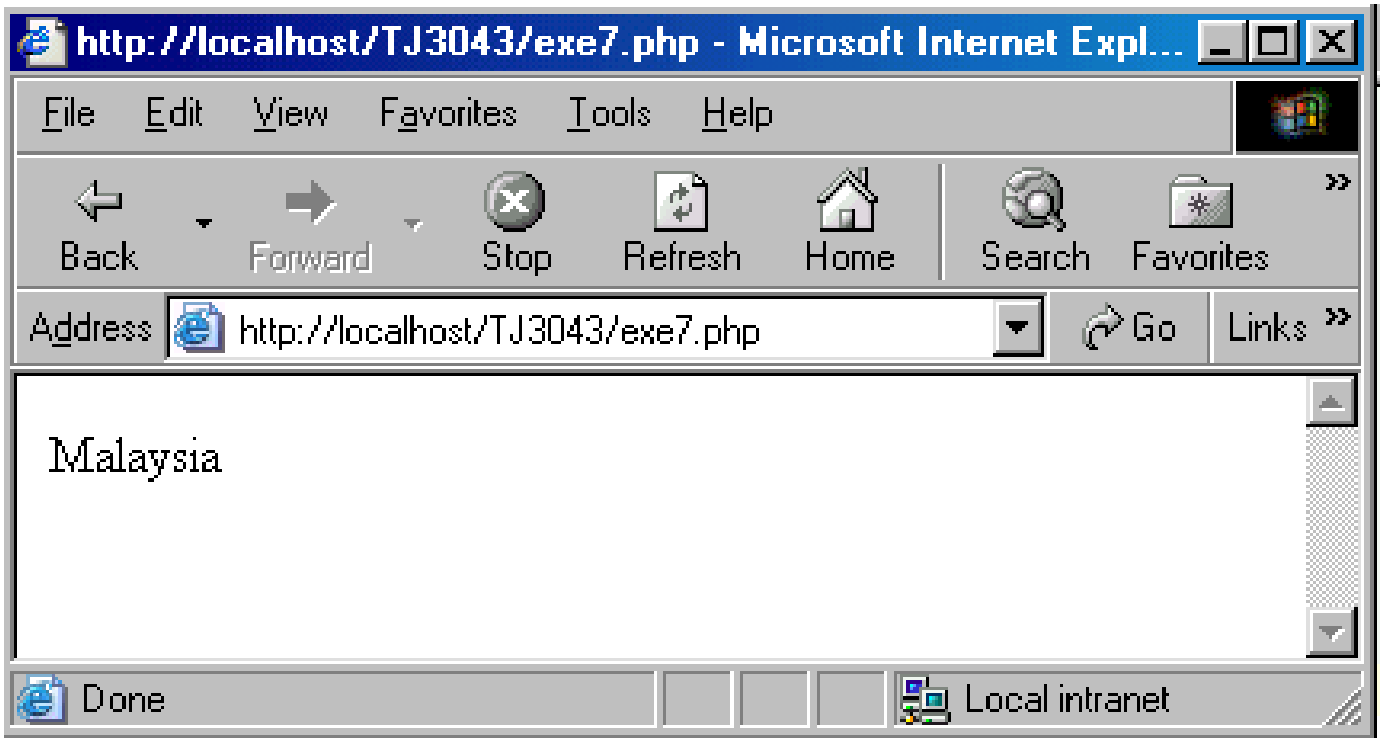
- ***Switch statement***
  - The code associated with the matching label will be executed or, if none of the labels match, the statement will execute any specified default code.
  - ***case*** keyword denote a label and the ***default*** keyword specify the default code
  - All label code must be terminated by a ***break*** statement

- Save as exe7.php

```
<?php
$country = "MY";

switch($country)
 {
       case "MY":
      echo "Malaysia";
      break;
       case "TH":
       echo "Thailand";
      break;
       default:
       echo "no country";  } ?>
```

Output 7


http://localhost/TJ3043/exe7.php - Microsoft Internet Expl...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe7.php
Go Links
Malaysia
Done
Local intranet

Conditional Statements and Loops

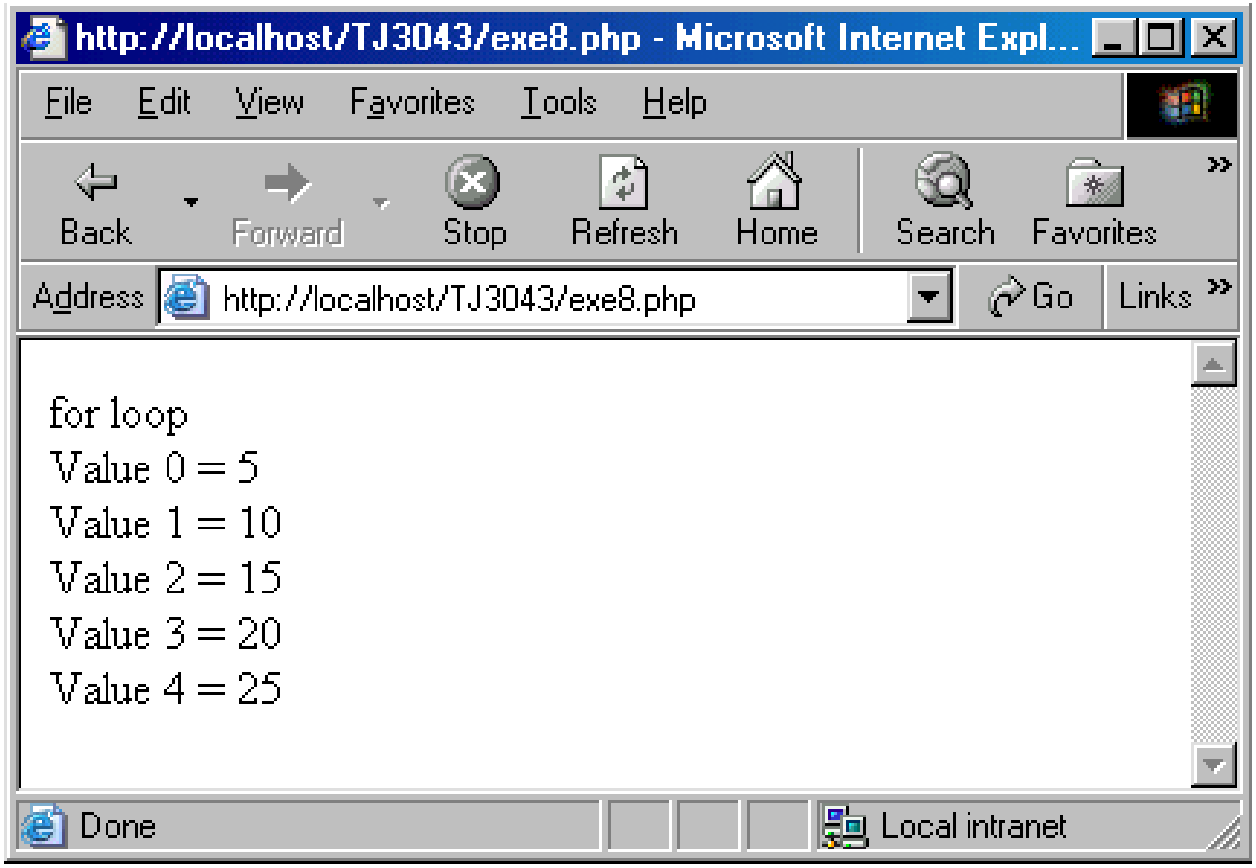
- For loop
  - Most frequently used type of loop in PHP scripting

- Save as exe8.php

```
<?php
$a = 0;
echo "for loop <br>";

for ($i=0; $i < 5; $i++)
{
        $a += 5;
        echo "Value $i = $a <br>";
}
?>
```

Output 8


http://localhost/TJ3043/exe8.php - Microsoft Internet Expl...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe8.php
Go Links
for loop
Value 0 = 5
Value 1 = 10
Value 2 = 15
Value 3 = 20
Value 4 = 25
Done
Local intranet

How `for` loop works?

$i	$i++	$a	$a+5	$i < 5	Echo $i = $a
0		0	5	0 < 5	Value 0 = 5
	1		10	1 < 5	Value 1 = 10
	2		15	2 < 5	Value 2 = 15
	3		20	3 < 5	Value 3 = 20
	4		25	4 < 5	Value 4 = 25

```
<?php
$a = 0;
echo "for loop <br>";
for ($i=0; $i < 5; $i++)
{
$a += 5;
echo "Value $i = $a <br>";
```

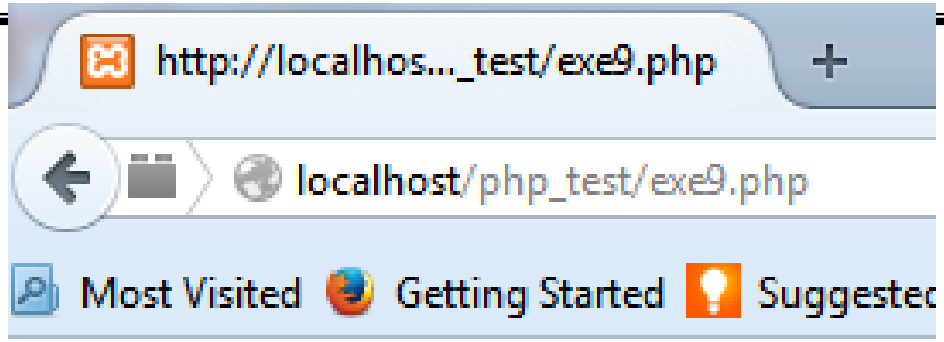
Conditional Statements and Loops

- While loop
  - *while* keyword followed by an expression to be evaluated for a Boolean value of *true* or *false*.

- Save as exe9.php

```
<?php
echo "while loop <br>";
$x = 0;
$a = 7;
while($x <5 )
{     echo "Value $a <br>";
      $x++;
      $a--;
}
echo "<p> Loop stopped at value $x of x";
?>
```

Output 9



while loop
Value 7
Value 6
Value 5
Value 4
Value 3

Loop stopped at value 5 of x

How `while` loop works?

$x	$x++	$a	$a--	$x < 5	Echo $a
0		7		0 < 5	Value 7
	1		6	1 < 5	Value 6
	2		5	2 < 5	Value 5
	3		4	3 < 5	Value 4
	4		3	4 < 5	Value 3

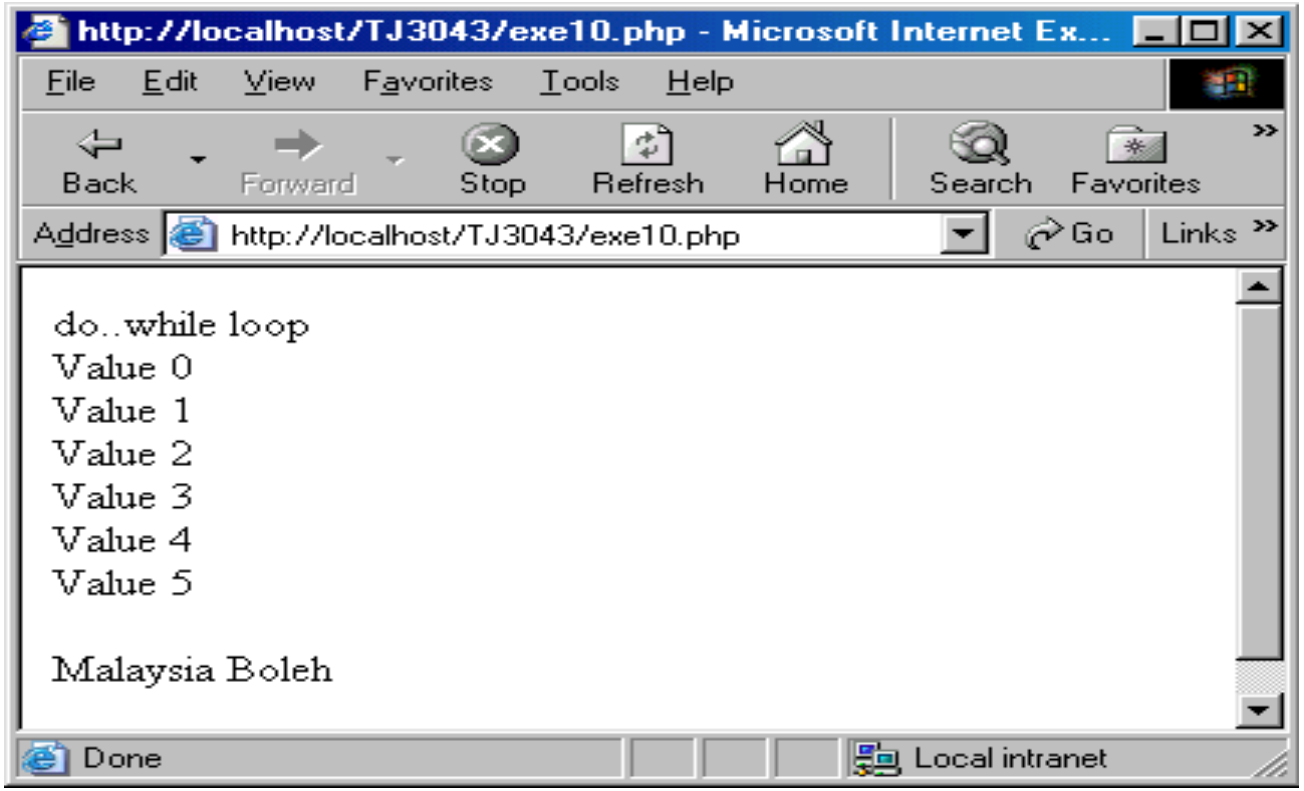
Conditional Statements and Loops

- Do-while loop
  - the loop runs at least once.
  - once the expressions evaluates as ***false***, the loop stops executing.

- Save as exe10.php

```
<?php
echo "do..while loop <br>";
$i = 0;
do { echo "Value $i <br>" ;}
while ($i++ < 5);
echo "<p>";
echo "Malaysia Boleh <p>";
?>
```

Output 10


http://localhost/TJ3043/exe10.php - Microsoft Internet Ex...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe10.php
Go Links
do..while loop
Value 0
Value 1
Value 2
Value 3
Value 4
Value 5
Malaysia Boleh
Done
Local intranet

How `do-while` loop works?

$i	$i++	$i < 5	Echo $i
0	(do statement)	NA	Value 0
	0	0 < 5	Value 1
	1	1 < 5	Value 2
	2	2 < 5	Value 3
	3	3 < 5	Value 4
	4	4 < 5	Value 5

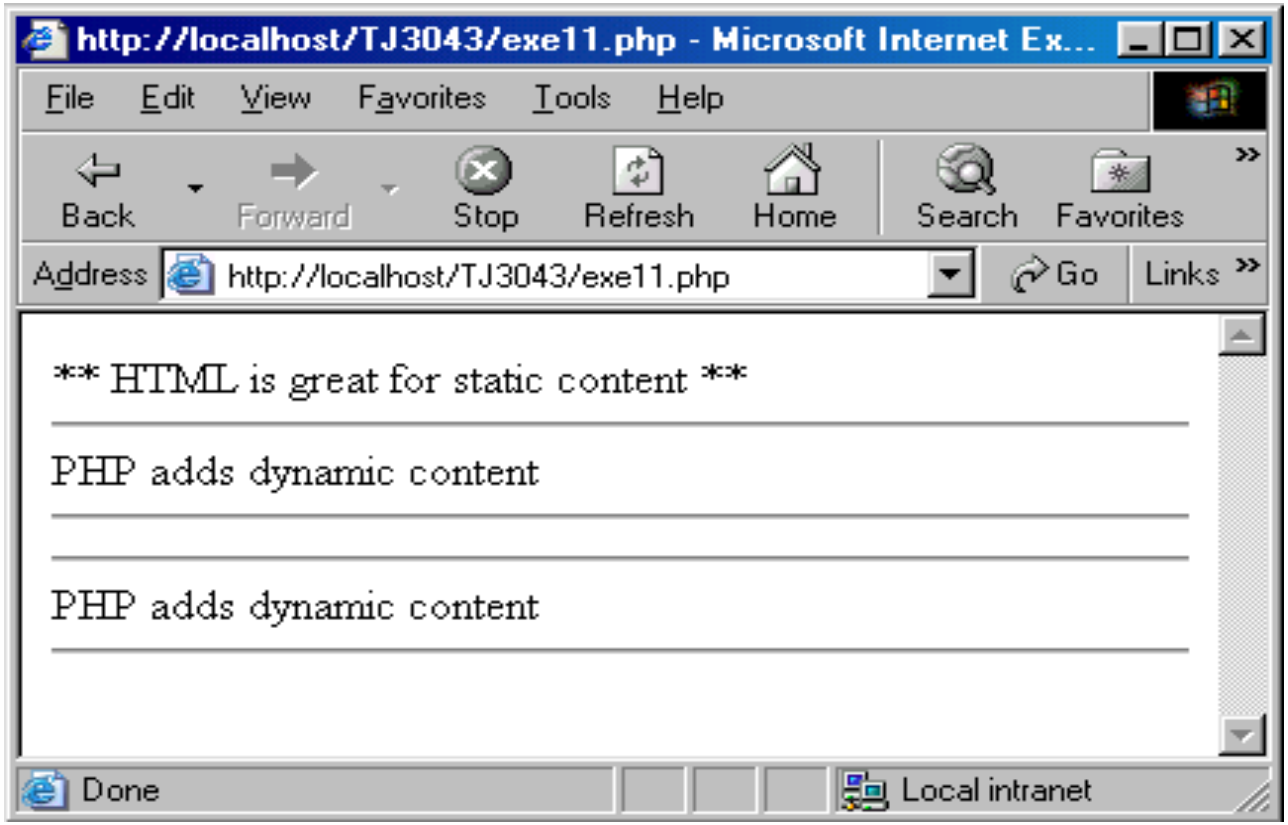
Functions

- A function is a piece of PHP code that perform a useful task, and usually return a value related to that task.
- It can be executed once or many times by the PHP script.

- Save as exe11.php

```
<?php
        function go()
      {
          echo"PHP adds dynamic content<hr>";
      }
echo "** HTML is great for static content **";
echo "<hr>";
go();
echo "<hr>";
go();      ?>
```

Output 11


http://localhost/TJ3043/exe11.php - Microsoft Internet Ex...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe11.php
Go Links
** HTML is great for static content **
PHP adds dynamic content
PHP adds dynamic content
Done
Local intranet

Function Argument

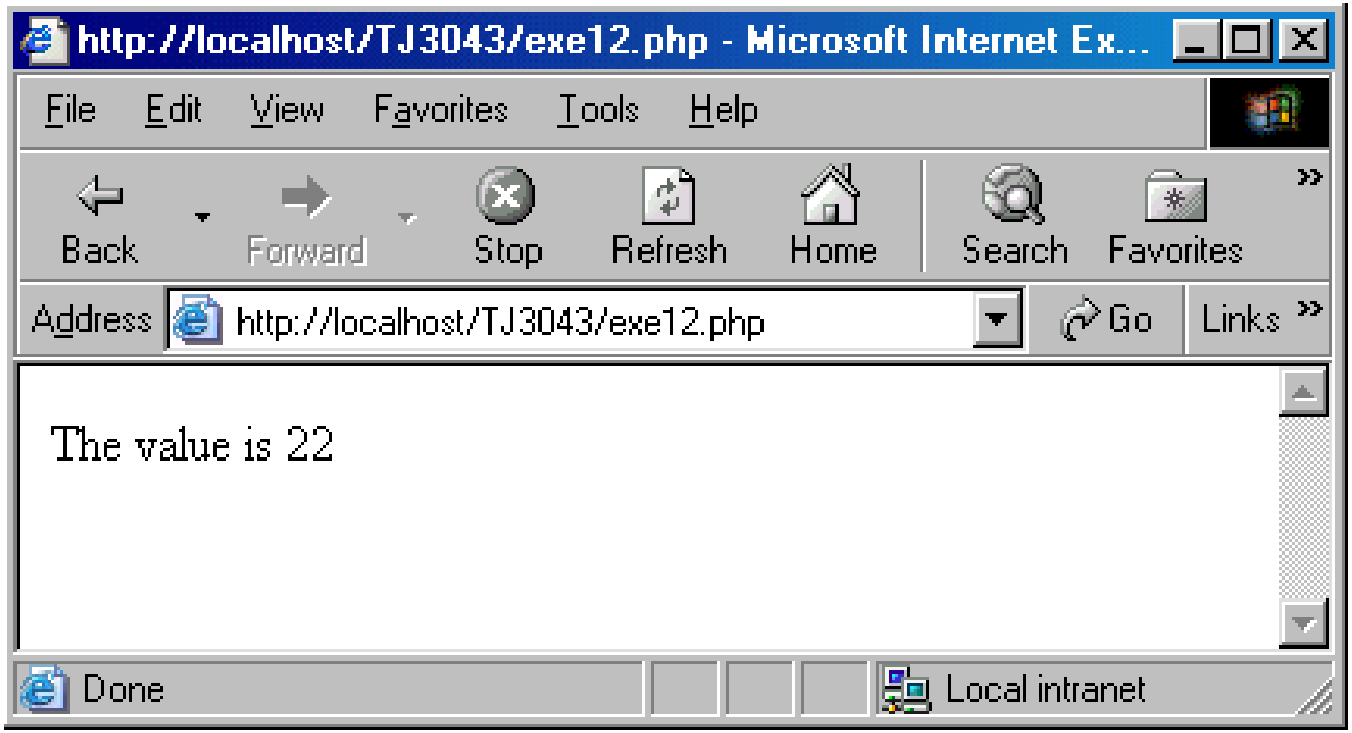
- The data contained within the brackets that follow the function names is known as the function ***argument***..

- Save as exe12.php

```
<?php
	function make_triple($arg1, $arg2, $arg3)
	{
		$num = $arg1 + $arg2 + $arg3;
		return $num;
	}

$value=make_triple(9,7,6);
echo("The value is $value");
?>
```

Output 12


http://localhost/TJ3043/exe12.php - Microsoft Internet Ex...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://localhost/TJ3043/exe12.php
Go Links
The value is 22
Done
Local intranet

ARRAY

What is an Array?

- A variable is a storage area holding a number or text. The problem is, a variable will hold only one value.
- An array is a special variable, which can store multiple values in one single variable.
- If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
$cars1="Saab";
$cars2="Volvo";
$cars3="BMW";
```

- An array can hold all your variable values under a single name. And you can access the values by referring to the array name.
- Each element in the array has its own index so that it can be easily accessed.

ARRAY

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.

1. In the following example the index are automatically assigned (the index starts at 0):

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

2. In the following example we assign the index manually:

```
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
```

NUMERIC ARRAYS

- In the following example you access the variable values by referring to the array name and index:

```
<?php
$cars[0]="Proton Exora";
$cars[1]="Volvo";
$cars[2]="Inokom";
$cars[3]="Toyota";
echo $cars[0] . " and " . $cars[2] . " are
Malaysian cars.";
?>
```

- The code above will output:

```
Proton Exora and Inokom are Malaysian cars.
```

ASSOCIATIVE ARRAYS

- An associative array, each ID key is associated with a value.
- When storing data about specific named values, a numerical array is not always the best way to do it.
- With associative arrays we can use the values as keys and assign values to them.

Example

- In this example we use an array to assign ages to the different persons:

```
$ages = array("Aiman"=>32, "Aisyah"=>30, "Ruhul"=>34);
```

ASSOCIATIVE ARRAYS

Example 2

- This example is the same as example 1, but shows a different way of creating the array:

```
$ages['Aiman'] = "32";
$ages['Aisyah'] = "30";
$ages['Ruhul'] = "34";
```

ASSOCIATIVE ARRAYS

- The ID keys can be used in a script:

```
<?php
$ages[‘Aiman'] = "32";
$ages[‘Aisyah'] = "30";
$ages[‘Ruhul'] = "34";

echo “Aiman is " . $ages[‘Aiman'] . " years old.";
?>
```

- The code above will output:

```
Aiman is 32 years old.
```

MULTIDIMENSIONAL ARRAYS

- In a multidimensional array, each element in the main array can also be an array. And each element in the sub-array can be an array, and so on.

MULTIDIMENSIONAL ARRAYS

Example

- In this example we create a multidimensional array, with automatically assigned ID keys:

```
$families =array(
    "Affendi"=>array("Aiman","Aisyah","Ruhul"),
    "Hamidi"=>array("Haikal"),
    "Sabri"=>array("Ikram","Noraini Sabrina")
  );
```

MULTIDIMENSIONAL ARRAYS

- Lets try displaying a single value from the array above:

```
echo "Is " . $families['Affendi'][2] .
" a part of the Affendi family?";
```

- The code above will output:

```
Is Ruhul a part of the Affendi family?
```

THE FOREACH LOOP

- The `foreach` loop is used to loop through arrays.
- Syntax

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

- For every loop iteration, the value of the current array element is assigned to $value (and the array pointer is moved by one) - so on the next loop iteration, you'll be looking at the next array value.

THE FOREACH LOOP

Example

- The following example demonstrates a loop that will print the values of the given array:

```
<html>
<body>
<?php
$x=array("one","two","three");
foreach ($x as $value)
  {
  echo $value . "<br />";
  }
?>
</body>
</html>
```

- Output:

```
one
two
three
```