Web Engineering

BY

Adnan Amin

IMSciences|Peshawar

Outlines

- What is PHP and What can PHP do?
- Hello World!
- PHP Basics
- Variable naming rules
- Arithmetic Operators in PHP
- Assignment Operators
- Comparison Operators
- Logical Operators
- String concatenation

What is PHP and What can PHP do?

- PHP: Hypertext Preprocessor
- Widely used open source general purpose scripting language
- Designed especially for web development
- Easily embedded into HTML
- PHP can do;
  - Collect form data
  - Generate dynamic page content
  - Send and receive cookies
  - Working with session
  - Working with file & folders

Hello World!

```
1. <?php
2.         echo “Hello world”;
3. ?>
```

- The entire php code should be placed inside <?php ….. ?> tags.
- <?php is starting of php section
- ?> is ending of php section
- Multiple php code can be placed into a single HTML document
- Extension of php file should be *.php

PHP Basics

- //This is single line comment
- /* this is multiline comments */
- Semicolons (;) should always be placed in the end of each statement
  - E.g. $x=111;
- The $ symbol: always be placed in front of all variables in php.

Variable naming rules

1. When creating PHP variable, must follow these three rules:
2. Variable names must start with a letter of the alphabet or the underscore character.
3. Variable names may not contain spaces. If a variable must comprise more than one word, it should be separated with underscore character e.g. $first_name etc.
4. Variable names are case-sensitive. The variable $first_name is not the same as the variable $First_Name.

Arithmetic Operators in PHP

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

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`

String concatenation

- Period (.) is used to append one string of characters to another.
- E.g. echo “Hello Dear “ . $user_name . “ Thanks”;

Conditional Statements

- It alter the program normal.
- It enable you to ask questions about certain things and respond to the answers you get in different ways.
- allow your program to choose different paths of execution based upon the outcome of an expression or the state of a variable.
- These statements allow you to control the flow of your program’s
- execution based upon conditions known only during run time.
- If statement

Non-Looping (Selection) and Looping (iteration) Conditional Statements

- Conditionals can be in the form of looping or non-looping.
- Non-looping conditional statements are;
  - The actions initiated by the statement take place and program flow then moves on.
  - If statement
  - Switch statement
  - ? Operator
- Looping conditional statements are;
  - It execute code over and over until a condition has been met.
  - For loop
  - While loop
  - Foreach loop

If Statement

- It evaluate the condition (i.e. expression).
- The actions to take when an if condition is TRUE are generally placed inside curly braces, { }.
- There is no need of braces if the true statement body contains single line of code.

```
IF(condition)
{
    Body for True case
}
```

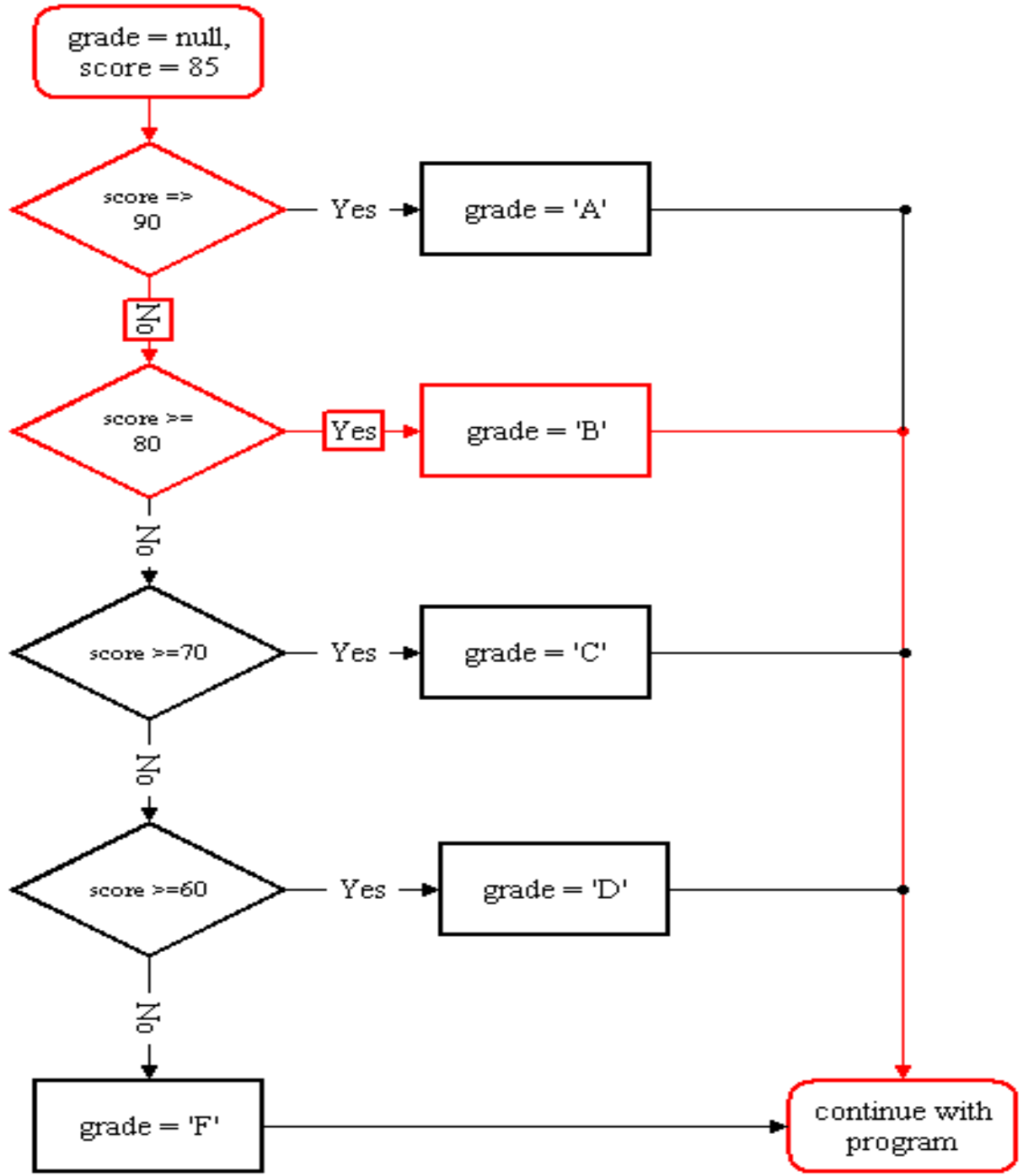
IF-Else Statement

- When a condition is true then body of if-part will be executed but when a condition is false then body of else-part will be executed.
- Sometimes when a conditional is not TRUE, you may not want to continue on to the main program code immediately but might wish to do something else instead. This is where the else statement comes in.

```
if(Condition)
{
        Body of If-Part
}
else
{
        Body of else-part
}
```

If-Else Statement

```
$grade;
$score = 85;
if ($score >= 90)
        $grade = 'A';
else if ($score >= 80)
        $grade = 'B';
else if ($score >= 70)
        $grade = 'C';
else if ($score >= 60)
        $grade = 'D';
else
        $grade = 'F';
```

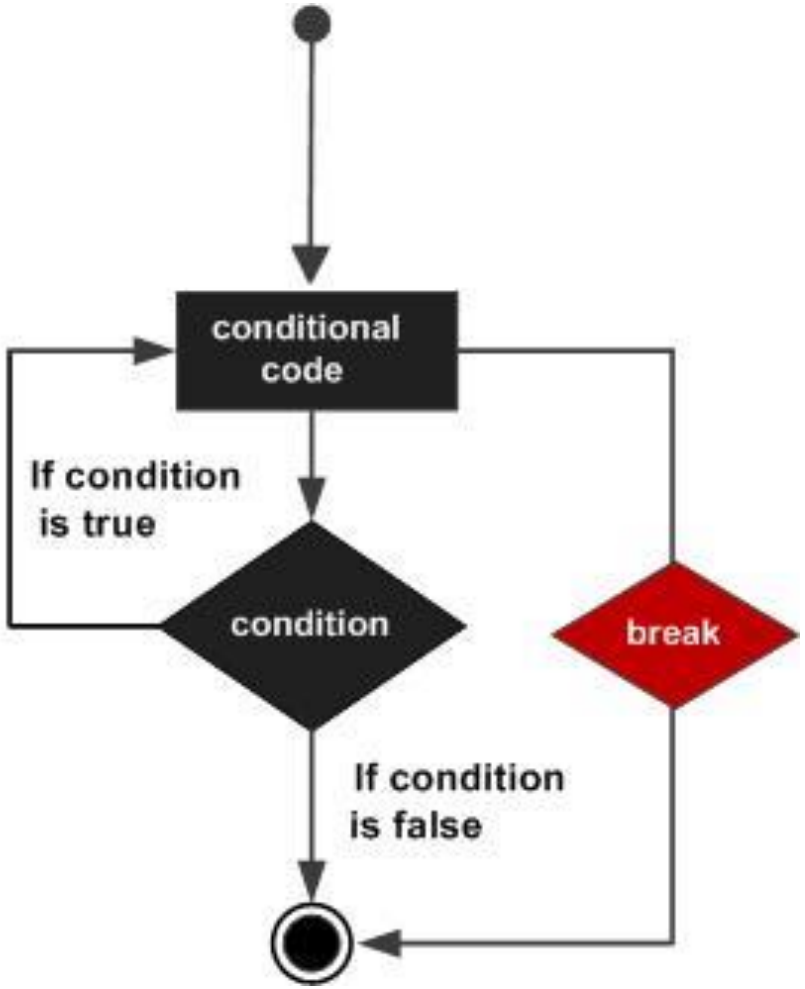


The switch Statement

- The switch statement is useful in cases in which one variable or the result of an expression can have multiple values, which should each trigger a different function.
- It is Java's multiway branch statement.

It works like this;

- ✓ The value of the expression is compared with each of the values in the case statements.
- ✓ If a match is found, the code sequence following that case Statement is executed.
- ✓ If none of the constants matches the value of the expression, then the Default statement (optional) is executed.
- ✓ If no case matches and no default is present, then no further action is taken.
- ✓ The break statement is used inside the switch to terminate a Statement sequence.



A multiple-line if ... elseif ... statement

```
1. <?php
2. if ($page == "Home")       echo "You selected Home";
3. elseif ($page == "About") echo "You selected About";
4. elseif ($page == "News")  echo "You selected News";
5. elseif ($page == "Login")  echo "You selected Login";
6. elseif ($page == "Links")   echo "You selected Links";
7. ?>
```

A switch statement

```
1.  <?php
2.  switch ($page)
3.  {
4.  case "Home":
5.  echo "You selected Home";  break;
6.  case "About":
7.  echo "You selected About";  break;
8.  case "News":
9.  echo "You selected News";   break;
10. case "Login":
11. echo "You selected Login";   break;
12. case "Links":
13. echo "You selected Links";    break;
14. }
15. ?>
```

The ? Operator

- The ? operator is passed an expression that it must evaluate, along with two statements to execute:
- One for when the expression evaluates to TRUE, the other for when it is FALSE.

```
<?php
        echo $fuel <= 1 ? "Fill tank now" : "There's enough fuel";
?>
```

Assigning a ? conditional result to a variable

Here $enough will be assigned the value TRUE only when there is more than a gallon of fuel; otherwise, it is assigned the value FALSE.

```
<?php
    $enough = $fuel <= 1 ? FALSE : TRUE;
?>
```

Thanks!