SOEN 287
WEB PROGRAMMING

Introduction to PHP - 1

Origins of PHP

http://freshnaukri.com/php-interview-answer-question-rasmus-lerdorf/

- **Origins**
  - Rasmus Lerdorf a Greenlandic programmer with Canadian citizenship created the PHP scripting language to allow him to track visitors to his Web site.
  - authoring the first two versions of the language
  - participated in the development of later versions led by a group of developers.

What is PHP?

http://www.w3schools.com/php/php_intro.asp

- PHP was originally an acronym for *Personal Home Page*, but later it became *PHP: Hypertext Preprocessor*
- PHP is a widely-used, open source scripting language

http://www.thetechnicalstuff.com/checking-php-installation-using-phpinfo-function/

PHP vs. JavaScript

- When a browser finds *JavaScript* code embedded in an HTML document it is displaying, it calls its JavaScript interpreter to interpret the script.

- When a browser requests a document that includes *PHP* script, the Web server that provides the document calls its PHP processor

- The server determines that a document includes PHP script by the file -name extension.

PHP Interpretation

- PHP files have extension `.php`
- PHP processor takes PHP document file as input and produces an HTML document file.
- The PHP processor has two modes:
  - **Copy mode**: copies markup code (include embedded client -side script) to the output file.
  - **Interpret mode**: executes PHP script and sends any output of the script to the output file.

PHP Interpretation

- PHP code are executed on the server, and the result is returned to the browser as plain HTML (!DOCTYPE html)

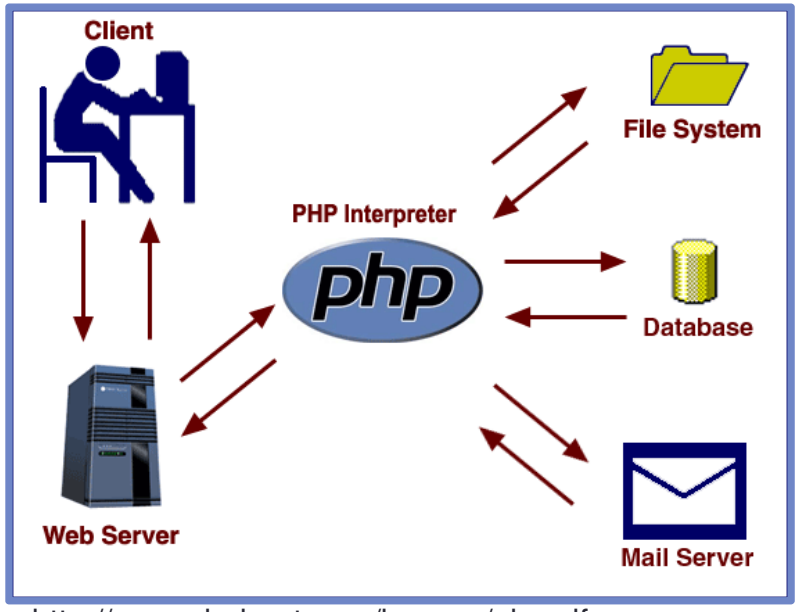
- The client never sees the PHP script (view source) just what is returned.
  (will see an example in a few slides)

http://lerablog.org/wp-content/uploads/2013/05/php.jpg

What can PHP do?

http://www.w3schools.com/php/php_intro.asp

- can generate dynamic page content
- can create, open, read, write, and close files on the server
- can collect form data
- can send and receive cookies
- can add, delete, modify data in your database
- can restrict users to access some pages on your website
- can encrypt data



http://www.shahnet.com/images/php.gif

Why is PHP used?

1. Easy to use & easy to learn
2. Runs on various platforms (Windows, Linux, Unix, …)
3. Compatible with most servers (Apache, …)
4. Supports a wide range of databases such as MySQL
5. Costs nothing, it is free to download and use

General Syntactic Characteristics

- PHP code can be specified in an HTML document
  - *Internally*: The PHP must be imbedded in HTML markup document by including it between following tags

```
<?php
   ...
   ?>
```

  - *Externally*: if a PHP script is stored in a different file, it can be brought into a document with the include construct:

```
include ("myScript.inc");
```

  - (We will see later)

- The file can have both PHP and HTML; can jump in and out of PHP "mode"

General Syntactic Characteristics

- If the file has PHP, the PHP must be in

  `<?php .. ?>`,

  even if the `include` is already in

  `<?php .. ?>`

View Source of : echo.php

An example of PHP:

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

PHP Comments

http://www.w3schools.com/php/php_syntax.asp

- *Comments* - three different kinds (Java and C)

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

# This is also a single line comment


/*
This is a multiple lines comment block
that spans over more than
one line
*/
?>
```

PHP Variables

- No type declaration – dynamic typing

- Variable name always begin with `$`
- Variable name can contain letters, digits, underscore
- Variable name cannot start with a digit.
- Variable names are case sensitive

- An unassigned (unbound) variable has the value, NULL
  - ✓The **`unset`** function sets a variable to NULL
  - ✓The **`isset`** function is used to determine whether a variable is NULL (`isset($aVar)` returns `true` of `false`)

Some Reserved words in PHP

and	else	global	require	virtual
break	elseif	if	return	xor
case	extends	include	static	while
class	false	list	switch	
continue	for	new	this	
default	foreach	not	true	
do	function	or	var	

NOTE:
Although variable names are case sensitive, reserved word and functions are not!

PHP Primitive Types

Integer

Double

- 3.45
- .345
- 345e3
- 345E-5

http://curtiscenter.math.ucla.edu/content/are-there-more-fractions-or-decimals

PHP Primitive Types …

https://wiki.openjdk.java.net/display/OpenJFX/Virtual Keyboard+User+Experience+Documentation

String (no character type)

- Defined with ' ' or " "
- Difference:
  Say have `$name = "Anna";`
  - The value of `'My name is $name'` is
    `My name is` **`$name`**
  - The value of `"My name is $name"` is
    `My name is` **`Anna`**
- With " will substitute value of variable (interpolation) while with ' will not

PHP Primitive Types …

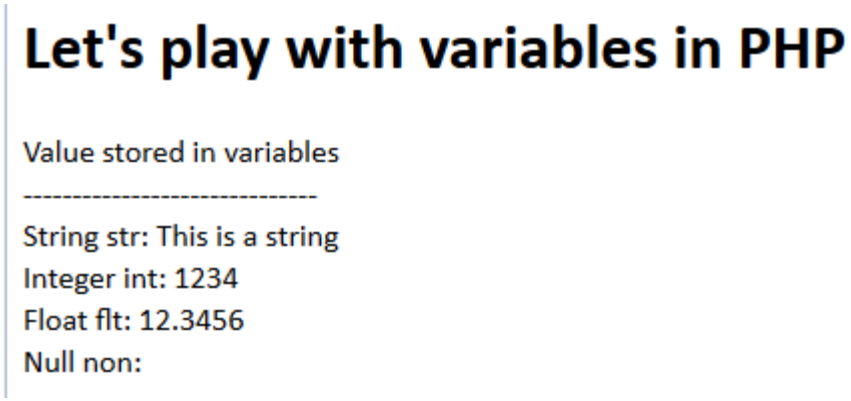
Boolean

- Possible values `FALSE` and `TRUE` (case sensitive)
- What if have a non-Boolean in Boolean context?
  - If an integer, 0 evaluates to `FALSE` otherwise it is `TRUE`
  - If a string, the empty string or the string “0” evaluates to `FALSE` otherwise it is `TRUE`
  - The string "0.0" evaluates to …..?
  - If a double, 0.0 exactly evaluates to `FALSE` otherwise to `TRUE`. Be careful with rounding errors.

PHP Variables - Example

```
<?php
    $str = "This is a string";
    $int = 1234; $flt = 12.3456;
    $non = NULL;
    // NULL means no value
    echo "String str: $str<br />";
    echo "Integer int: $int<br />";
    echo "Float flt: $flt<br />";
    echo "Null non: $non<br />";

?>
```



variables.php

PHP Variables – Example …

```
<?php
  $yes = isset($str);
  echo "Is the string variable assigned a value? $yes<br />";
  unset($str);
  $yes = isset($str);
  echo " Now? $yes<br />";

?>
```

```
Demonstrating unset and isset
---------------------------------
Is the string variable assigned a value? 1
Now?
```

variables.php

PHP Arithmetic Operators

Operator	Operations
+	Numeric addition
.	String concatenation
-	Subtraction
*	Multiplication
/	division
%	Modulus*
++	Increment
--	Decrement

```
<?php
 $addnum=1+2;
 $addstr="php"." programming";
 $inc = 5;
 $inc = ++$inc;
 $dec = $inc--;

 $result = "addnum:$addnum<br />";
 $result .="addstr:$addstr<br />";
 $result .="inc:$inc<br />";
 $result .="dec:$dec<br />";
?>
```

Output:

* The modulus operator (%) coerces its operands to integer, if necessary

PHP Logical Operators

Operator	Operations
&&	Logical AND
and	Logical AND
||	Logical OR
or	Logical OR
xor	Logical exclusive OR
!	Logical NOT

```
<?php
   echo (true and true)? "true":
              "false";
?>
```

Output:

PHP Logical Operators

Operator	Operations
&&	Logical AND
and	Logical AND
||	Logical OR
or	Logical OR
xor	Logical exclusive OR
!	Logical NOT

```
<?php
   echo (true or true)? "true":
              "false";
?>
```

Output:

PHP Logical Operators

Operator	Operations
&&	Logical AND
and	Logical AND
||	Logical OR
or	Logical OR
xor	Logical exclusive OR
!	Logical NOT

```
<?php
  echo (true xor true)? "true":
          "false";
?>
```

Output:

PHP conditionals - Example

```
<?php
   $num = 8;
   if($num%2 == 0)
       $msg= "$num is an even number";
   else
       $msg = "$num is an odd number";
   echo ($msg);
?>
```

Output:

Loops in PHP – Same as Java

while loop:

```
<?php
    $i=0; $num=50;
    while ($i < 10)
    {
       $num--;
       $i++;
    }
    echo ("Loop stopped at $i<br />\$num is
now $num");
?>
```

Output

```
Loop stopped at 10
$num is now 40
```

Loops in PHP – Same as Java …

do … while loop:

Output

```
Loop stopped at 1
$num is now 49
```

```
<?php
   $i=0; $num=50;
   do {
      $num--;
      $i++;
   } while ($i<1);
   echo ("Loop stopped at $i<br />\$num
is now $num");
?>
```

Loops in PHP – Same as Java …

for loop:

Output

```
at the end of the loop a=50 and b=25
```

```
<?php
    $a=0; $b=0;
    for($i=0;$i<5;$i++)
    {
      $a +=10;
      $b += 5;
    }
    echo("at the end of the loop a=$a and
b=$b");
?>
```

String operations in PHP

- `$name='John Smith';  or`
  `$name="John smith";`

- `$first= "John"; $last="Smith";`
  `$name="$first $last";`

- Two strings can be concatenated together with the dot operator (`.`).

```
$me = 'It\'s my name ' . $first . " $last\n";
```

PHP string functions

strlen (*str*)	Returns the number of characters in *str*.
trim (*str*) **trim** (*str, charlist)* **ltrim** (*str)* **rtrim** (*str)*	Returns a string by stripping white space (or other characters) from both ends of *str.*
substr (*str, i* [,*len*])	Returns a substring of *str* from position *i* (zero-based indexing) to the end or to length *len*
strstr (*line, word)*	Finds first instance of a string *word* in the longer string *line* and returns a substring of *line* that begins with *word*.

Converting String to array using explode()

- **Example:**
- **$str1= "Saturday, Sunday, Monday"**
- **$array= explode(",", $str1)**

- **If the string elements are separated by "," split them and save each element in an array.**

- **This can be helpful for file applications.**

Just Checking

Given the code

```
<?php
  $str = "Hello World!";
  echo $str . "<br>";
  echo trim($str,"Hed!");
?>
```

What is the output?

PHP string functions

strtolower (*s*) **strtoupper** (*s*)	Returns the lowercase, uppercase version of *s*.
strcmp (*str_1*, *str_2*)	Returns a positive, negative, or zero integer if *str_1* is greater than, less than, or equal to *str_2*.
ucwords (s)	Returns a given string with the first letter of each word in uppercase

Just Checking -

What is output?

```
<body>
<?php    $first = ‘john...';    $last = “Smith";
   echo   (trim($first,".")."<br />");
   echo   (strlen($first)."<br />");
   echo   (strstr(“John smith",“mi").'<br />');
   echo   (ucwords(“john SMITH").'<br />');
   echo   (strcmp("Hello", "hello"));
?>
</body>
```

stringFunctions.php

Scalar Type Conversion (check)

- **Scalar type: boolean,integer, float, string**
- **Implicit** - coercions
  - Context of expression determines type that is expected or required
  - *String to numeric (if string in a numeric context)*
    - If the string contains an e or an E (1.3e3) , it is converted as well.
    - If the string does not begin with a sign or a digit, zero is used

Example

- ```
  <?php
  $foo = 1 + "10.5";                // $foo is float (11.5)
  $foo = 1 + "-1.3e3";              // $foo is float (-1299)
  $foo = 1 + "bob-1.3e3";           // $foo is integer (1)
  $foo = 1 + "bob3";                // $foo is integer (1)
  $foo = 1 + "10 Small Pigs";       // $foo is integer (11)
  $foo = 4 + "10.2 Little Peggies"; // $foo is float (14.2)
  $foo = "10.0 pigs " + 1;          // $foo is float (11)
  $foo = "10.0 pigs " + 1.0;        // $foo is float (11)
  ?>
  ```
  ```

Scalar Type Conversion

- **Explicit** – casts

  e.g., `(int)$total` or `intval($total)` or `settype($total, "integer")`

- The type of a variable can be determined with `gettype` or `is_type`

  `gettype($total)` - it may return `"unknown"`

  `is_integer($total)` – a predicate function (`is_boolean()`, `is_string()` .....)

Assignment Operators

- Simple assignment (=)
- Compound assignment
  - +=
  - −=
  - /=
  - etc. ….

Output statements

- Output from a PHP script becomes part of HTML document that is sent to the browser so all output must be in HTML or XHMTL format

- Two ways to produce output: `print` and `printf`

  1. `print` takes a string, but will coerce other values to strings

     - `print "This is too <br /> much fun <br />";`
     - `print 72;`

Output statements

2. printf

```
$price = 100;
$item = "desk";
printf("The price of %s is %d", $item, $price);
```

Example – if1.php

```
<h1>Try to roll a six</h1>
  <p>
    <a href = "if1.php">roll again</a>
  </p>
  <?php
     $roll = rand(1,6);
     if ($roll == 6){
       print("<h1>Holy Guacamole! That's a
six!</h1>\n");
     }
     else
    {
      print("Roll is ".$roll);
    }
 ?>
```

Example – if2.php

```
<?php
   $roll = rand(1,6);
   print <<<HERE
 <p>
   <img src = "images/dado_$roll.png"
         alt = "$roll"
         height = "100px"
         width = "100px" />
 </p>
HERE;
  if ($roll == 6){
   print("<h1>Holy Guacamole! That's a
six!</h1>\n");
  } // end if
?>
```

HTML5 and CSS3 All-in-one for Dummies, Andy Harris

Arrays

- Not like the arrays of any other programming language
- A PHP array is a generalization of the arrays of other languages
- To create an array use the `array()` construct, which takes one or more key `=>` value pairs as parameters and returns an array of them
- The keys are non-negative integer literals or string literals
- The values can be anything

Arrays …

- A “regular” array of strings

```
$list = array(0 => "apples",
        1 => "oranges", 2 => "grapes")
```

- If a key is omitted, 0 is the default key
- If more than a key is omitted and there have been integer keys, the default key will be the largest current key + 1. (Repeat the same process for the following keys)
- If a key appears that has already appeared, the new value will overwrite the old one

Example:

```
<?php
 $list = array(5, 9 => 7, "year" => 10, 8, 11);
    foreach( $list as $key=>$value)
        echo " Key=" . $key . ", Value=" . $value;
 ?>
```

Result:

Key=0, Value=5 Key=9, Value=7 Key=year, Value=10
Key=10, Value=8 Key=11, Value=11

Examples of Creating an Array

- ```
  $list = array("make" => "Cessna",
      "model" => "C210","year" => 1960,
      3 => "sold");
  ```

- ```
  $list = array(1, 3, 5, 7, 9);
  ```

- ```
  $list = array(5, 3 => 7, 5 => 10,
      "month" => "May");
  ```

- ```
  $colors = array('red', 'blue', 'green',
      'yellow');
  ```

Examples of Creating an Array

Create then fill

```
<?php    $arr=array();
         $arr[0]="First";
         $arr[1]=" PHP";
         $arr[2]=" array";
         echo ($arr[0].$arr[1].$arr[2]);
?>
//-------------------------------------
```

Create and fill at the same time

```
<?php    $mo=array("Jan ","Feb ","Mar ");
         $dy= array("25 ","26 ","27 ");
         $yr=array("2012","2013","2014");
         echo  ($mo[1].$dy[0].$yr[2]);
?>
```

Arrays in PHP – Creating …

Index can be a string – Associative arrays

```
<?php   $b =array("first_name"=>"John",
                  "last_name"=>“Smith");
        $b[’email’]="john.smith@gmail.com"; ?>

//-------------------------------------
```

Looping through an array

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

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

http://www.w3schools.com/Php/php_arrays.asp

Looping through an associative array

```
<?php
     $price=array("Apples"=>"2.99",
"Banana"=>"0.79", "Joe"=>"43");
foreach($price as $x=>$x_value)
{
     echo "Key=" . $x . ", Value=" .
$x_value;
     echo "<br />"; }
?>
```

```
Key=Apples, Value=2.99
Key=Banana, Value=0.79
Key=Joe, Value=43
```

PHP array functions

Function	Action
`count($ar)`	Returns the number of entries in `$ar`
`empty($ar)`	Returns true if `$ar` is
`array_pop($ar)`	Pops (deletes) and returns last entry, or `null`
`array_push($ar,$e1,$e2,...)`	Inserts one or more elements to the end of `$ar`
`array_shift($ar)`	Pops (deletes) and returns first entry, or `null`

PHP array functions

Function	Action
`array_unshift($ar,$e1,...)`	Inserts one or more elements at the beginning of `$ar`
`array_reverse($ar)`	Returns an array in the reversed order
`sort($ar)` `rsort($ar)`	Sorts array in increasing or decreasing order (array keys will reset)
`ksort($ar)` `krsort($ar)`	Sorts array keys in increasing or decreasing order
`asort($ar)` `arsort($ar)`	Sort associative arrays in increasing or descending order, according to the value (array keys remain same as original)