PHP
Variables, Data Types & Flow Controls

Lecture 4

Table of Content

Variables

- It consists a name preceded by a dollar sign ($).
  Syntax: `$name;`
- `name` include letters, numbers and underscore character (_).
- No spaces is allowed.
- Must start with letter or underscore.

Variables cont.

- No declaration is needed to define a variable.
- Variable names are case sensitive.

Local, Global and Superglobal

- All users variables are local to their script or function.
- Global variables will be discussed with using functions.
- PHP had predefined variables called superglobals.

Superglobal Variables

- $_GET
  contains variables provided by GET method
- $_POST
  contains variables provided by POST method
- $_COOKIE
  contains variables provided through a cookie

Superglobal Variables cont.

- $_FILES
  contains variables provided through file uploads
- $_SERVER
  contains info. Such as headers, file paths and script locations.
- $_ENV
  contains variables provided as part of server environment.

Superglobal Variables cont.

- $_REQUEST
  contains variables provided through any user input mechanism.
- $_SESSION
  contains variables currently registered in a session.

All these superglobals will be discussed in details later.

Table of Content

- Variables
- Data Type
- Operators & Expressions
- Constants
- Flow Controls

Data Types

- A variable could be used as string variable then as an int variable in the same script.
- There are eight standard data types:

Data Types cont.

- Data Types:
  - `bool` `true` `false`
  - `integer` `5` `1987`
  - `float or double` `3.23` `15.458`
  - `string` “Hi” “Welcome”
  - Object
  - Array index or associative arrays
  - Resource reference third-party resource
  - `NULL` uninitialized variable

Changing Variable Type

- By using `settype()` function
  - Syntax:
    ```
    settype($variableName, ‘newType’);
    ```
  - Example:
    ```
    $varA = 3.14;
    settype($varA, ‘integer’);
    print(“New value = ”.$varA);
    ```

Changing Variable Type cont.

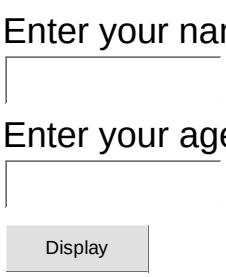
- By using Type Casting
  - Syntax:
    ```
    $newVar = (newType)$orignialVar
    ```
  - Example:
    ```
    $varA = 3.14;
    $varB = (integer) $varA
    print("New value = ".$varB);
    ```


Changing Variable Type cont.

- Why is changing type useful?
  - Example:
    ```
    $in1 = "30cm";
    $in2 = "50cm3m";
    $f1 = (integer)$in1; //30
    $f2 = (integer)$in2; //50
    echo "You entered:".$f1."x".$f2;
    ```

Example

- Build a form that will accept the user age in term of years and print it on the browser.



Example Solution – form.html

```
<html>
  <head><title>Form</title></head>
  <body>
     <form action="1.php" method="post">
          Enter your name:
          <input type="text" id="name"><br>
          Enter your age:
          <input type="text" id="age"><br>
          <input type="submit">
     </form>
  </body>
</html>
```

Example Solution – 1.php

```
<html>
  <head><title>Form Output</title></head>
  <body>
     <?php
          echo “Your name is ”.$_POST[name];
          echo “”<br/>””;
          settype($_POST[age], “integer”);
          echo “Your age is ”.$_POST[age];
     ?>
  </body>
</htm>
```


Table of Content

- Variables
- Data Type
- Operators & Expressions
- Constants
- Flow Controls

Operators and Expressions

- Arithmetic operators
  - +, - , *, /, %, ++, --
- Concatenation operator
  - . Used to concatenate between to value to produce a string

Operators and Expressions cont.

- Assignment operators
  - =, +=, -=, *=, %=, .=
- Comparison operators
  - ==, !=, >, <, >=, <=
- Logical operators
  - or, ||, and, &&, !

Operator Precedence

++, --, (cast)	right
*, /, %	left
+, -, .	left
<, <=, >, >=	nonassociative
==, !=, ===	nonassociative
&&	left
||	left
? :	left
=, +=, -=, *=, %=, .=	right
AND	left
XOR	left
OR	left

Example

- Build a web page that contain a form which needs the price of a unit, quantity and the discount amount. Then it will display the result in same page.

Price:
Quantity:
Discount:

Calculate

Total:

Example Solution – cal.php

```
<html>
  <head><title>Calculator</title></head>
  <body>
     <form action="cal.php" method="post">
          Price:
          <input type="text" id="price"><br>
          Quantity:
          <input type="text" id="quan"><br>
          Discount:
          <input type="text" id="dis"><br>
          <input type="submit">
```


Example Solution – cal.php cont.

```
     </form>
     <?php
          $total = $_POST[price]*$_POST[quan];
          $total -= $total*$_POST[dis]/100;
          echo "<b>Total:</b>";
          echo $total." Saudi Riyals.";
     ?>
  </body>
</htm>
```

Table of Content

- Variables
- Data Type
- Operators & Expressions
- Constants
- Flow Controls

Constants

- Defining a constant:

  Syntax:

  ```
  define("name", value);
  ```

  Example:

  ```
  define("Discount", 10);
  echo Discount;
  ```

Constants cont.

- You don't need to precede the constant name with $.
- If we want to make it case insensitive, we just send a 3rd boolean value true.

  Example:

```
define("Discount", 10, true);
echo Discount;
echo DISCOUNT;
```


Predefined Constants

- __FILE__

  returns the current file name.

- __LINE__

  returns the current line

Table of Content

- Variables
- Data Type
- Operators & Expressions
- Constants
- Flow Controls

Flow Controls

- Sequence
- Branching
  - if statement
  - if … else statement
  - if …elseif statement
  - switch statement
  - ? operator
- Repetition
  - while stat.
  - do … while stat.
  - for stat.
  - foreach stat.

Branching Statement Syntax

```
if(expression)
{ //statements if true
}
else
{ //statements if false
}
```


Branching Statement Syntax cont.

```
if(expression)
{ //statements if this exp=true
}
elseif(expression)
{ //statements if this exp=true
}
```

Branching Statement Syntax cont.

```
switch(expression)
{ case valueN:
        //stat. if exp=valueN
        break;
   default:
        //stat. if exp!= any
        //above value
}
```


Branching Statement Syntax cont.

```
$var = (exp)?trueVal:falseVal;
```

Repetition Statement Syntax

- ```
  while(expression)
  { //stat. if exp=true
  }
  ```

- ```
  do
  { //stat. if exp=true
  }while(expression);
  ```


Repetition Statement Syntax cont.

- ```
  for(initial; expr; update)
  { //stat. if exp=true
  }
  ```

- ```
  foreach(array as index=>value)
  { /* used with arrays. It will
       loop for each element and
       assign its index to index and
       its value to value.*/
  }
  ```

Break and Continue

- break;
  will exit from the current loop block
- continue;
  will skip the current iteration to the next one.

Example

- Build a page that will print a welcoming message with the appropriate title before the user name according to his/her gender.

Last name:

Gender: ◉Female ○Male

Display

Example Solution – wel.php

```
<html>
  <head><title>Welcome</title></head>
  <body>
     <form action="wel.php" method="post">
          Last name:
          <input type="text" id="name"><br>
          Gender:
          <input type="radio" id="gen"
          value="f">Female
          <input type="radio" id="gen"
          value="m">Male<br>
          <input type="submit">
```


Example Solution – wel.php cont.

```
     </form>
     <?php
          echo "Welcome ";
          if($POST[gen]=="f")
               echo "Ms.";
          else
               echo "Mr.";
          echo $_POST[name]." to our site";
     ?>
  </body>
</htm>
```