

# IT4552 – Web programming

## Chapter 2. PHP Variables and HTML Input Forms

1

### Objectives

- ❖ To learn how to store and access data in PHP variables
- ❖ To understand how to create and manipulate numeric and string variables
- ❖ To review how to create HTML input forms
- ❖ To learn how to pass data from HTML forms to PHP scripts

2

### Content

1. PHP Variables
2. Working with PHP String Variables
3. HTML Input Forms
4. HTML Input Forms and PHP Scripts

3

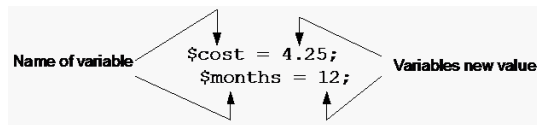
### Content

1. PHP Variables
2. Working with PHP String Variables
3. HTML Input Forms
4. HTML Input Forms and PHP Scripts

4

## 1. PHP Variables

- ❖ Variables are used to store and access data in computer memory.
- ❖ A variable name is a label used within a script to refer to the data.



## 1.1. Assigning New Values to Variables

- ❖ You can assign new values to variables:  

```
$days = 3;  
$newdays = 100;  
$days = $newdays;
```
- ❖ At the end of these three lines, **\$days** and **\$newdays** both have values of 100.

## Selecting Variable Names

- ❖ You can select just about any set of characters for a variable name in PHP, but they must:
  - Use a dollar sign (\$) as the first character
  - Use a letter or an underscore character ( \_ ) as the second character.
- ❖ Note: Try to select variable names that help describe their function. For example \$counter is more descriptive than \$c or \$ctr.

## Combining Variables and the print Statement

- ❖ That is, to print out the value of \$x, write the following PHP statement:
  - ```
print ("x");
```
- ❖ The following code will output "Bryant is 6 years old".  

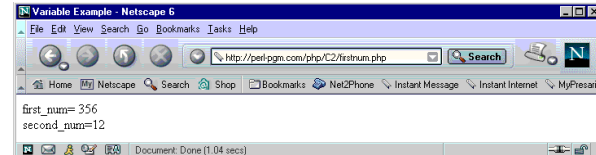
```
$age=6;  
print ("Bryant is $age years old.");
```

## A Full Example ...

```
1. <html>
2.   <head> <title>Variable Example </title> </head>
3.   <body>
4.     <?php
5.       $first_num = 12;
6.       $second_num = 356;
7.       $temp = $first_num;
8.       $first_num = $second_num;
9.       $second_num = $temp;
10.      print ("first_num= $first_num <br>
11.            second_num=$second_num");
12.    ?> </body> </html>
```

## A Full Example ...

The previous code can be executed:



## 1.2. Using Arithmetic Operators

❖ You can use operators such as a plus sign (+) for addition and a minus sign (–) for subtraction to build mathematical expressions.

❖ For example

```
<?php
$apples = 12;
$oranges = 14;
$total_fruit = $apples + $oranges;
print ("The total number of fruit is
      $total_fruit");
?>
```

❖ These PHP statements would output “The total number of fruit is 26.”

## Common PHP Numeric Operators

| Operator | Effect         | Example                                              | Result                           |
|----------|----------------|------------------------------------------------------|----------------------------------|
| +        | Addition       | <code>\$x = 2 + 2;</code>                            | <code>\$x</code> is assigned 4.  |
| –        | Subtraction    | <code>\$y = 3;</code><br><code>\$y = \$y – 1;</code> | <code>\$y</code> is assigned 2.  |
| /        | Division       | <code>\$y = 14 / 2;</code>                           | <code>\$y</code> is assigned 7.  |
| *        | Multiplication | <code>\$z = 4;</code><br><code>\$y = \$z * 4;</code> | <code>\$y</code> is assigned 16. |
| %        | Remainder      | <code>\$y = 14 % 3;</code>                           | <code>\$y</code> is assigned 2.  |

```

1. <html>
2. <head> <title>Variable Example </title> </head>
3. <body>
4. <?php
5. $columns = 20;
6. $rows = 12;
7. $total_seats = $rows * $columns;
8.
9. $ticket_cost = 3.75;
10. $total_revenue = $total_seats * $ticket_cost;
11.
12. $building_cost = 300;
13. $profit = $total_revenue - $building_cost;
14.
15. print ("Total Seats are $total_seats <br>");
16. print ("Total Revenue is $total_revenue <br>");
17. print ("Total Profit is $profit");
18. ?> </body> </html>

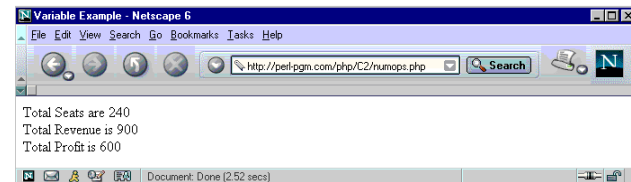
```

## A Full Example

13

## A Full Example ...

The previous code can be executed:



14

## WARNING: Using Variables with Undefined Values

A variable that does not have a value assigned to it will have no value (called a null value). When a variable with a null value is used in an expression PHP, PHP may *not* generate an error and may complete the expression evaluation.

```

<?php
$y = 3;
$y=$y + $x + 1; // $x has a null value
print ("x=$x y=$y");
?>

```

Output x= y=4

15

## 1.3. Writing Complex Expressions

- ❖ *Operator precedence rules* define the order in which the operators are evaluated. For example,  

$$x = 5 + 2 * 6;$$
- ❖ The value of  $x$  is either 42 or 17 depending on order of evaluation.
- ❖ Since multiplication evaluated before addition operations, this expression evaluates to 17.

16

## PHP Precedence Rules

- ❖ PHP follows the precedence rules listed below.
  - First it evaluates operators within parentheses.
  - Next it evaluates multiplication and division operators.
  - Finally it evaluates addition and subtraction operators.

17

## PHP Precedence Rules

- ❖ For example, the first 2 statements evaluate to 80 while the last to 180.
  - $\$x = 100 - 10 * 2;$
  - $\$y = 100 - (10 * 2);$
  - $\$z = (100 - 10) * 2;$

18

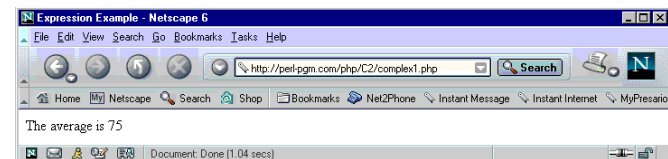
## A Full Example

```
1. <html>
2. <head> <title>Expression Example </title> </head>
3. <body>
4. <?php
5. $grade1 = 50;
6. $grade2 = 100;
7. $grade3 = 75;
8. $average = ($grade1 + $grade2 + $grade3) / 3;
9. print ("The average is $average");
10. ?> </body> </html>
```

19

## A Full Example ...

The previous code can be executed:



20

## Content

1. PHP Variables
2. Working with PHP String Variables
3. HTML Input Forms
4. HTML Input Forms and PHP Scripts

21

## 2. Working with PHP String Variables

- ❖ Character strings are used in scripts to hold data such as customer names, addresses, product names, and descriptions.
- ❖ Consider the following example.
  - `$name="Christopher";`
  - `$preference="Milk Shake";`
- ❖ `$name` is assigned "Christopher" and the variable `$preference` is assigned "Milk Shake".

22

## WARNING: Be Careful Not to Mix Variable Types

- ❖ Be careful not to mix string and numeric variable types.
- ❖ For example, you might expect the following statements to generate an error message, but they will not. Instead, they will output "y=1".

```
<?php
    $x ="banana";
    $sum = 1 + $x;
    print ("y=$sum");
?>
```

23

## Using the Concatenate Operator

- ❖ The concatenate operator combines two separate string variables into one.
- ❖ For example,
  - `$fullname = $firstname . $lastname;`
- ❖ `$fullname` will receive the string values of `$firstname` and `$lastname` connected together.
- ❖ For example,

```
$firstname = "John";
$lastname = "Smith";
$fullname = $firstname . $lastname;
print ("Fullname=$fullname");
```

24

## TIP: An Easier Way to Concatenate Strings

- ❖ You can also use double quotation marks to create concatenation directly,
- ❖ For example,
  - `$Fullname2 = "$FirstName $LastName";`
  - This statement has the same effect as
  - `$Fullname2 = $FirstName . " " . $LastName;`



25

## The strlen() Function

- ❖ Most string functions require you to send them one or more arguments.
- ❖ Arguments are input values that functions use in the processing they do.
- ❖ Often functions return a value to the script based on the input arguments. For example

`$len = strlen($name);`  
Receives the number of characters in \$name      Variable or value to work with  
Name of function



26

## The strlen() Function Example

```
<?php
    $comments = "Good Job";
    $len = strlen($comments);
    print ("Length=$len");
?>
```

This PHP script would output "Length=8".



27

## The trim() Function

- ❖ This function removes any blank characters from the beginning and end of a string. For example, consider the following script:

```
<?php
    $in_name = " Joe Jackson ";
    $name = trim($in_name);
    print ("name=$name$name");
?>
```



28

## The strtolower() and strtoupper() Functions

- ❖ These functions return the input string in all uppercase or all lowercase letters, respectively.

- ❖ For example,

```
<?php
    $inquote = "Now Is The Time";
    $lower = strtolower($inquote);
    $upper = strtoupper($inquote);
    print ("upper=$upper lower=$lower");
?>
```

- ❖ The above would output "upper=NOW IS THE TIME lower=now is the time".

## The substr() Function

- Substr has the following general format:

Assign the extracted substring into this variable.  $\$part = \text{substr}(\$name, 0, 5);$  Starting position to start extraction from. Number of characters to extract. (If omitted it will continue to extract until the end of the string.)

Extract from this string variable.

## The substr() Function

- ❖ The substr() function enumerates character positions starting with 0(not 1)

- For example, in the string "Homer", the "H" would be position 0, the "o" would be position 1, the "m" position 2, and so on

- ❖ For example, the following would output "Month=12 Day=25".

```
<?php
    $date = "12/25/2002";
    $month = substr($date, 0, 2);
    $day = substr($date, 3, 2);
    print ("Month=$month Day=$day");
?>
```

## The substr() Function

- ❖ As another example, consider the following use of the substr() function

- It does not include the third argument (and thus returns a substring from the starting position to the end of the search string)

```
<?php
    $date = "12/25/2002";
    $year = substr($date, 6);
    print ("Year=$year");
?>
```

- ❖ → Output "Year=2002"



## Content

1. PHP Variables
2. Working with PHP String Variables
3. HTML Input Forms
4. HTML Input Forms and PHP Scripts



33

## HTML Form

- ❖ Controls for User Interaction in HTML
  - To enter information and submit to a server



34

## HTML Form Example

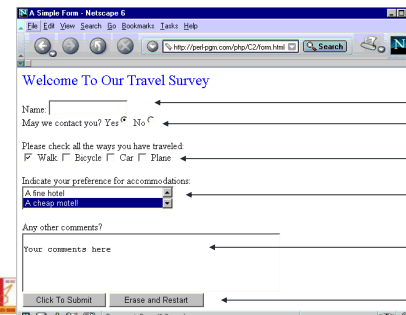
```
<form action="/test.php"
method="POST">
<p><input type="text"
name="username">
<input type="submit" value="Send"
></p>
</form>
```

35

## 3. HTML Input Forms

- ❖ HTML Forms and not part of PHP language but important way to send data to scripts



Text Box  
Radio Buttons  
Check Box  
Select Box  
Text Area

Submit/Reset button

36

### 3.1. Starting And Ending HTML Forms

- ❖ You can create HTML forms by using the HTML `<form>` and `</form>` tags

Program to start when form is submitted. →

```
<form action="http://webwizard.aw.com/~phpgm/program.php"
      method="post"> ← Format to send data.
```

▪

← Place form elements between `<form>` and `</form>` tags.

▪

`</form>` ← Forms end with `</form>`

37

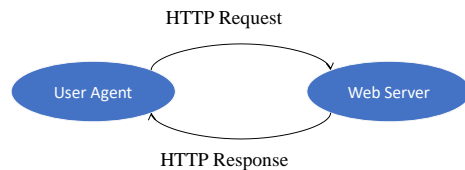
### HTML Form

- ❖ action attribute
  - URI Reference where you want to send data
- ❖ method attribute
  - Data transfer method
    - GET
      - Send data in the query part of the URI
    - POST
      - Send data in the body of the submission

38

### Review: Client Server Model (Web)

- ❖ Client: User Agent
- ❖ Server: Web server



39

### HTTP Request

	Method	URL	Protocol Version
	GET	/index.html	HTTP/1.1
Headers	Host: www.example.com		
	User-Agent: Mozilla/5.0		
	Accept: text/html, */*		
	Accept-Language: en-us		
	Accept-Charset: ISO-8859-1,utf-8		
Body (optional)	Connection: keep-alive		
	blank line		

40

## HTTP Response

Version      Status      Status Message

Headers

```
HTTP/1.1 200 OK
Date: Thu, 24 Jul 2008 17:36:27 GMT
Server: Apache-Coyote/1.1
Content-Type: text/html; charset=UTF-8
Content-Length: 1846
blank line
<html>
Body
...
</html>
```



## 3.2. Creating Form Buttons

- ❖ You can create submit and reset buttons by placing the following within `<form>` & `</form>` tags

```
<input type="submit" value="Click To Submit">
<input type="reset" value="Erase and Restart">
```

Type of button to create      Button Label

- ❖ The submit button will be labeled "Click To Submit". The reset button will be labeled "Erase and Restart".



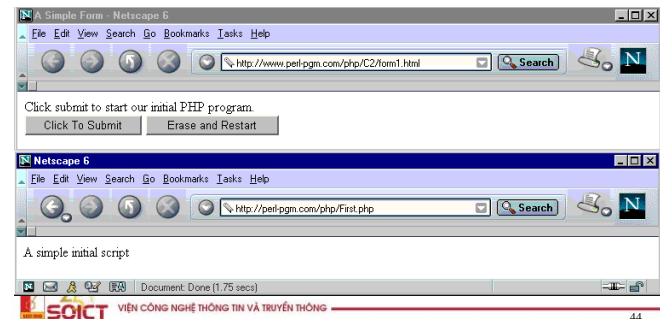
## Another Full Script Example

```
1.<html>
2.<head> <title> A Simple Form </title> </head>
3.<body>
4.<form
  action="http://webwizard.aw.com/~phppgm/First.php"
  method="post" >
5. Click submit to start our initial PHP program.
6. <br> <input type="submit" value="Click To Submit">
7. <input type="reset" value="Erase and Restart">
8. </form>
9. </body> </html>
```



## A Full Example ...

The previous code can be executed:



### 3.3. Creating Text Input Boxes

- ❖ Text input boxes create a form element for receiving a single line of text input.

Name: `<input type="text" size="15" maxlength="20" name="fname">`

Annotations:

- ↑ Create a text box.
- ↑ The width of text box.
- ↑ Maximum number of input characters
- ↑ Use this name to identify the form element in the receiving program.

- ❖ Will be 15 characters wide accepting a maximum of 20 characters. Will set a variable named `fname` with value of whatever the end-user enter.

### 3.4. Creating Password Boxes

- ❖ Password boxes similar to text boxes except asterisks are displayed (instead of text input).

`<input type="password" size="15" maxlength="20" name="pass1">`

Annotations:

- ↑ Create a password text box.
- ↑ The width of text box.
- ↑ Maximum number of input characters
- ↑ This variable name will be set in the receiving PHP script.

- ❖ Will be 15 characters wide accepting a maximum of 20 characters. Will set a variable named `pass1` with value of whatever the end-user enter.

### Warning: Password Boxes Not Secure

- ❖ When the user submits the form, any data input is sent in clear text (nonencrypted) just like any other HTML form field.
- ❖ Someone with network access could, therefore, read the password being transferred.
- ❖ For this reason, most Web applications do not use this approach to receive and transmit passwords.

### 3.5. Creating Text Areas

- ❖ The following creates a text area containing 4 rows and 50 columns.

`<textarea rows="4" cols="50" name="Comments">`  
Your comments here  
`</textarea>`

Annotations:

- ↑ Number of rows
- ↑ Number of columns.
- ↑ Text areas have closing tags.
- ↑ Any text here will appear as default text in text area.

- ❖ The words "Your comments here" are the default text. The variable name `Comments` will be available to the form-handling script.

### 3.6. Creating Radio Buttons

- ❖ Radio buttons are small circles that can select by clicking them with a mouse. Only one within a group can be selected at once.

```
<input type="radio" name="contact" value="Yes" checked>
<input type="radio" name="contact" value="No" >
```

Annotations:

- ↑ Create radio button.
- ↑ Since both radio buttons have the same name, the radio buttons will operator together.
- ↑ The value that will be sent to the form-processing program.
- ↑ This item will be pre-checked when the form is viewed.

- ❖ The name argument must be the same for all radio buttons operating together. The value argument sets the variable value that will be available to the form-processing script.

### 3.7. Creating Check Boxes

- ❖ Check boxes are small boxes on a form that create a check mark when the user clicks them.

```
<input type="checkbox" name="walk" value="Yes" checked> Walk
<input type="checkbox" name="Bicycle" value="Yes"> Bicycle
<input type="checkbox" name="Car" value="Yes"> Car
<input type="checkbox" name="Plane" value="Yes"> Plane
```

Annotations:

- ↑ Create checkbox
- ↑ Each check box sets a different variable name when selected.
- ↑ This item will be pre-checked when the form is viewed.
- ↑ The value that will be sent to the form-processing program.

- ❖ The above create four independent check boxes; that is, all four check box elements can be selected and each will set a value for a different variable name.

### 3.7. Creating Check Boxes (2)

- ❖ Might want to create a set of check boxes that use the same name argument.

```
<input type="checkbox" name="travel" value="Car" checked> Car?
<input type="checkbox" name="travel" value="Bike"> Bicycle?
<input type="checkbox" name="travel" value="Horse"> Horse?
<input type="checkbox" name="travel" value="None"> None of the above?
```

Annotations:

- ↑ Create checkbox
- ↑ Since each checkbox element has the same name, multiple values can be set for the same variable name.
- ↑ This item will be pre-checked when form is viewed.
- ↑ The value that will be sent to the form-processing program.

- ❖ The value received by the form-processing script would be a comma-separated list of all items checked.

### 3.8. Creating Selection Lists

- ❖ Creates a box with a scrolling list of one or more items that user can highlight and select.

```
<select name="Accommodations" size=2 multiple>
<option> A fine hotel </option>
<option selected> A cheap motel! </option>
<option> A tent in the parking lot </option>
<option> Just give me a sleeping bag checked </option>
</select>
```

Annotations:

- ↑ variable name set in the receiving script
- ↑ Viewable window size
- ↑ Allows end-user to select multiple items.
- ↑ This text is displayed as an option and the entire text will be returned as the variable's value if selected.

- ❖ This HTML code creates four options formatted in a scrolling list. Only two of these options are displayed at the same time, and the user can select more than one option. Multiple selections are sent to the form-processing script as a comma-separated list.

## Content

1. PHP Variables
2. Working with PHP String Variables
3. HTML Input Forms
4. HTML Input Forms and PHP Scripts

53

## Receiving Form Input into PHP Scripts

- ❖ To receive HTML form input into a PHP script:
  - Use a PHP var name that matches the variable defined in the form element's **name** argument.
- ❖ E.g., if form uses the following:
  - `<input type="radio" name="contact" value="Yes">`
- ❖ Then form-handling PHP script could use a variable called `$contact`.
  - If the user clicks the radio button, then `$contact` would = **Yes**

54

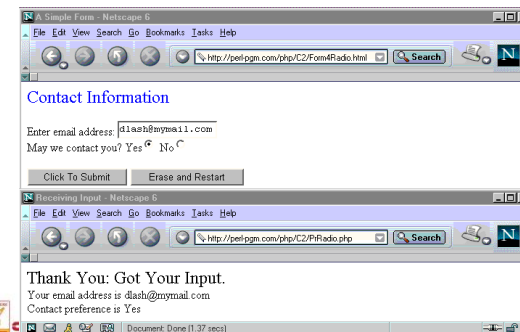
## Full Example

- ◆ Suppose your HTML form uses the following
  - ◆ Enter email address: `<input type="text" size="16" maxlength="20" name="email">`
- ◆ Then can receive input as follows
  1. `<html>`
  2. `<head><title> Receiving Input </title> </head>`
  3. `<body>`
  4. `<font size=5>Thank You: Got Your Input.</font>`
  5. `<?php`
  6. `print ("<br>Your email address is $email");`
  - 7.
  8. `print ("<br> Contact preference is $contact");`
  9. `?>`

55

## A Full Example ...

The previous code can be executed:



56

## Register\_Globals?

- ❖ Since PHP 4.2.1, the default PHP configuration is require a different mechanism to receive input for security reasons (than the one just shown).
  - Technical details: it is a PHP configuration option to turn REGISTER\_GLOBALS OFF (new default) or ON in the php.ini configuration file.
- ❖ If your site has REGISTER\_GLOBALS OFF you must use a different mechanism to receive HTML Form Variables.

## How can you tell if Register\_Globals is OFF?

- ❖ Enter the following PHP script and run it.
  - `<?PHP phpinfo(); ?>`
- ❖ Search through the output for REGISTER\_GLOBALS and see if it is set to OFF or ON.
- ❖ If it is off you must use the following way to receive input data.

## Getting input data with Register\_Globals OFF?

- ❖ To receive data with REGISTER\_GOBALS OFF you use a special variable called `$_POST`.

`$name $_POST["name"];`

Enclose in square bracket and then quotes

Name of HTML form variable (note do not use \$)

Special PHP Global variable. Technically it is an *associative array* (covered in chptr 5.)

PHP variable name that you want to receive the HTML form input.

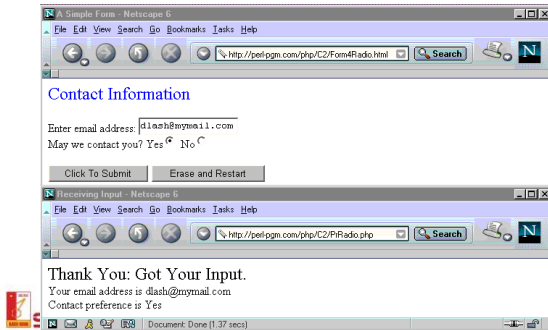
## Full Example, when REGISTER\_GLOBALS is OFF

- ◆ Suppose your HTML form uses the following email address: `<input type="text" size="16" maxlength="20" name="email">`
- ◆ Then can receive input as follows

```
1. <html>
2. <head><title> Receiving Input </title> </head>
3. <body>
4. <font size=5>Thank You: Got Your Input.</font>
5. <?php
6.     $email = $_POST["email"];
7.     $contact = $_POST["contact"];
8.     print ("<br>Your email address is $email");
9.     print ("<br> Contact preference is $contact");
10. ?>
```

## A Full Example ...

The previous code can be executed:



61

## Summary

- ❖ Variables are used to store and access data in computer memory.
- ❖ You can associate a value with a variable, change that value, print it out, and perform many different operations on it.
- ❖ PHP supports both numeric and string variables. String variables use different methods for value manipulation (for example, concatenation) than numeric variables do.

62

62

## Summary

- ❖ You can use HTML forms to pass data to PHP scripts. HTML form elements include text boxes, text areas, password boxes, check boxes, radio buttons, and selection lists.
- ❖ PHP scripts can receive form element input values by using a PHP variable name that matches the one specified in the form element's name argument.
  - If RESITER\_GLOBALS is off in your installation you must get input data using `$POST["var_name"]`;

63

63

## Exercise 1. Firefox and Add-ons

1. Install Firefox
  - <http://www.mozilla.com/en-US/firefox/>
2. Install Add-ons
  - DOM Inspector
  - Firebug
  - Web Developer
3. Restart Firefox

64



## Exercise 4

- ❖ Create a calculator form:

A simple calculator form. It consists of a top input field for the first number, a row of four buttons for operators (+, -, x, /), a second input field for the second number, and a 'Submit' button at the bottom right.

Result:

## Exercise 5

- ❖ Demonstration of string functions

A form for demonstrating string functions. It has a top input field, a row of four buttons for string functions (strlen, strtolower, trim, strtoupper), a second input field, and a 'Submit' button at the bottom right.

Result:

## Exercise 6

- ❖ Create a form as the following. Pressing on the submit button will display all input information

A screenshot of a web browser displaying a 'Welcome To Our Travel Survey' form. The form includes fields for Name, a contact preference question, a travel mode selection, an accommodation preference dropdown, and a text area for comments. It has 'Click To Submit' and 'Erase and Restart' buttons at the bottom.

## Question?

