

Chapter 2. PHP Variables and HTML Input Forms

Content

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

Content



1. PHP Variables

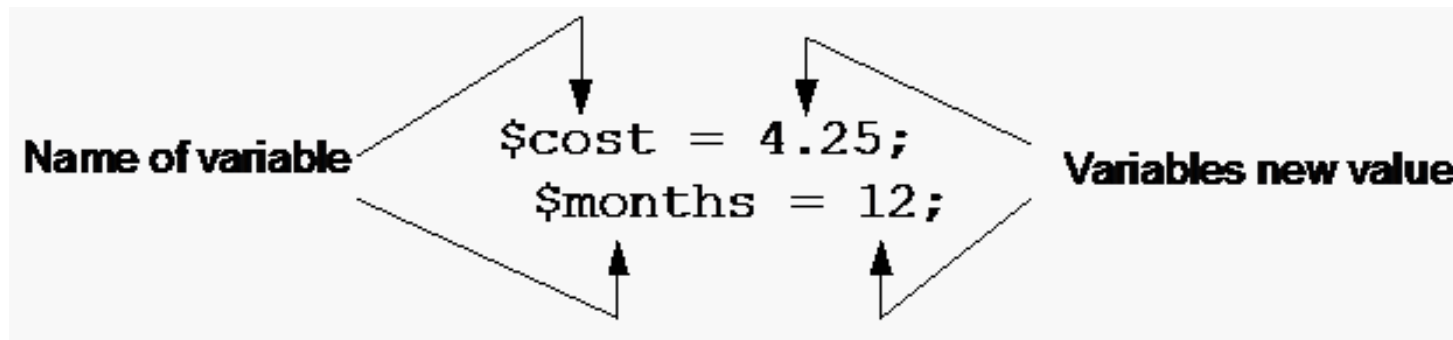
2. Working with PHP String Variables

3. HTML Input Forms

4. HTML Input Forms and PHP Scripts

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”.

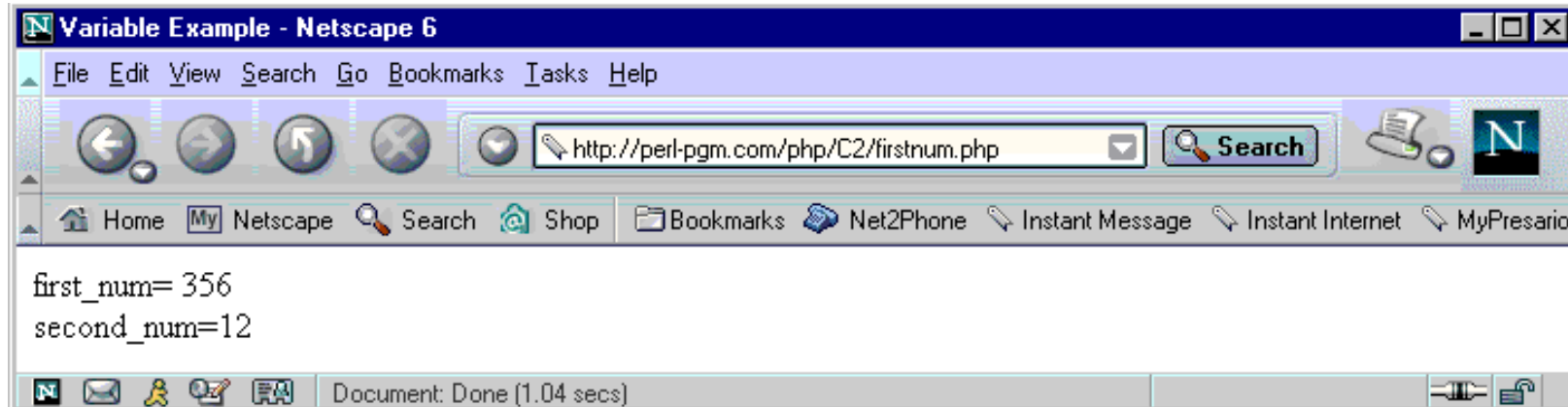
```
$age=6;
```

```
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= $first_num <br>
        second_num=$second_num");
11.    ?> </body> </html>
```


A Full Example ...



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.”

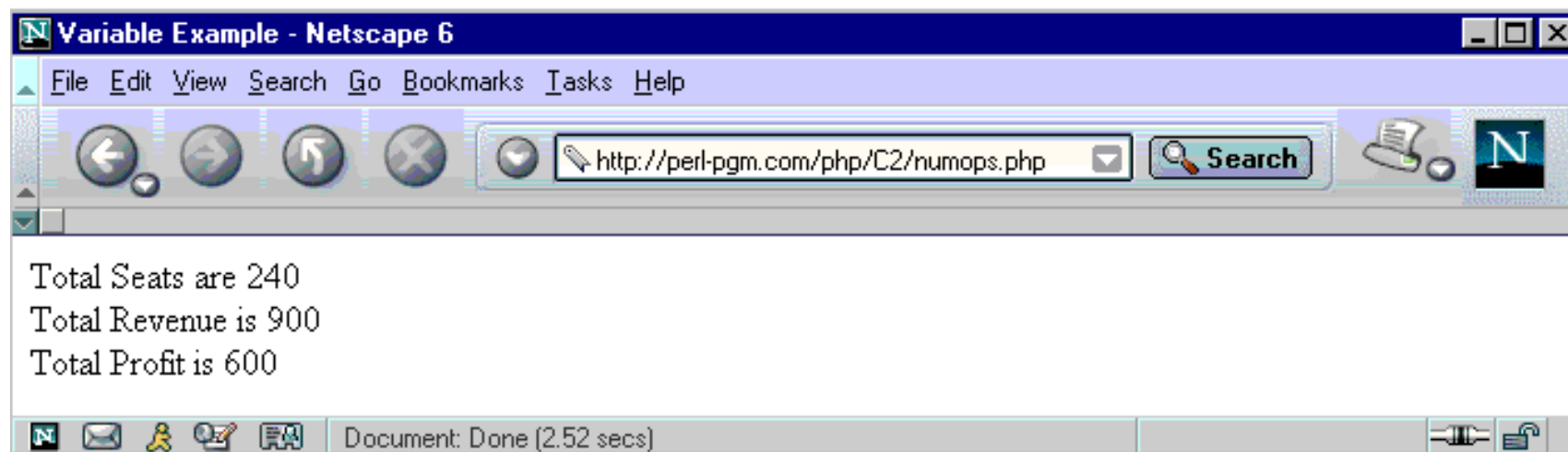
Common PHP Numeric Operators

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

A Full Example

```
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 ...



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

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.

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.

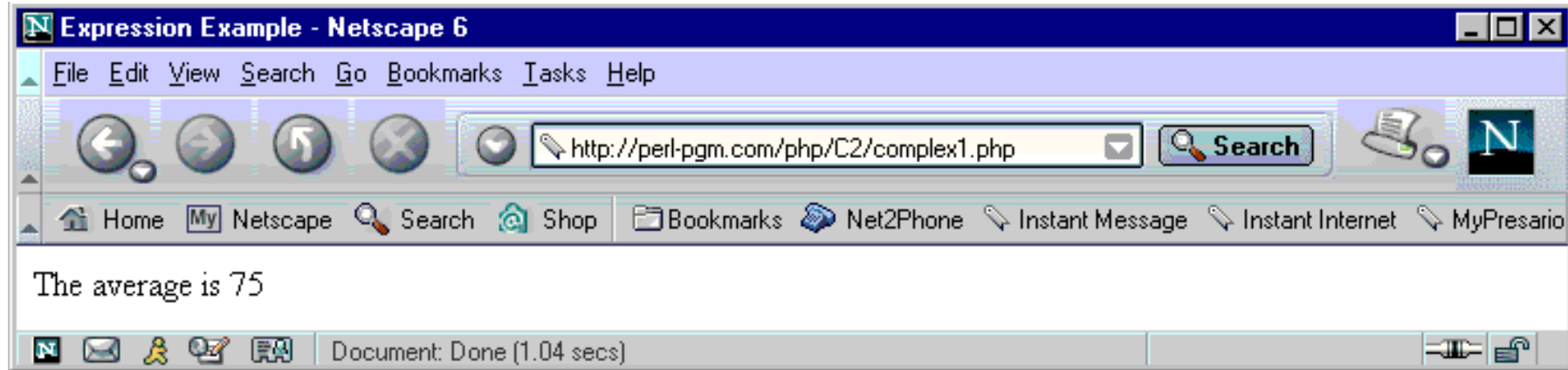
PHP Precedence Rules

- 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;`

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>
```

A Full Example ...



Content

1. PHP Variables

→ 2. Working with PHP String Variables

3. HTML Input Forms

4. HTML Input Forms and PHP Scripts

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”.

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");
```

```
?>
```

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");
```

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;`

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

Name of function

Variable or value to work with

The strlen() Function Example

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

This PHP script would output “Length=8”.

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");
?>
```

The strtolower() and strtoupper() Functions

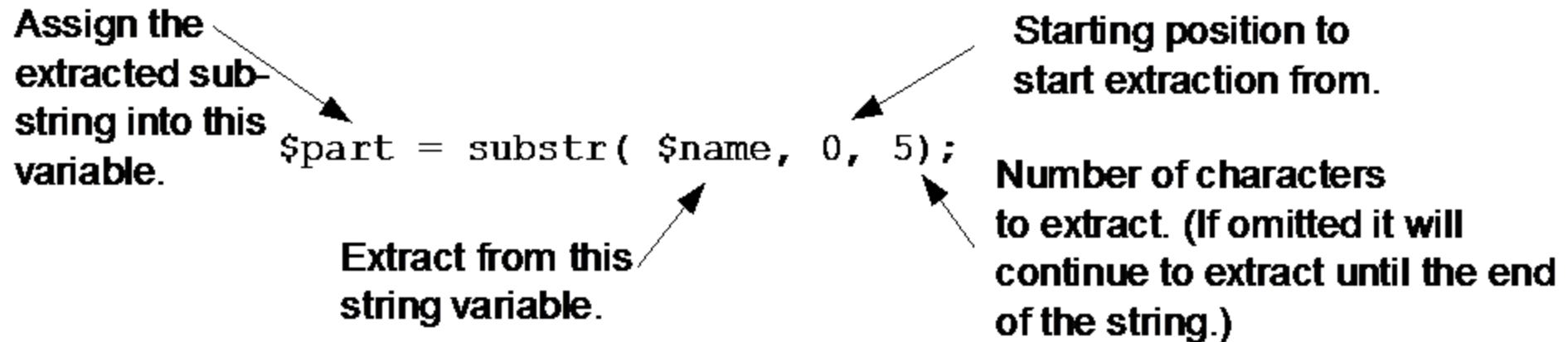
- These functions return the input string in all uppercase or all lowercase letters, respectively.
- For example,

```
<?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:



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

HTML Form

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

The image displays two web pages side-by-side. The left page is the Google Japan homepage, featuring the Google logo with a red and yellow key as the letter 'l'. Below the logo is a search bar with a blue border, and two buttons: 'Google Search' and 'I'm Feeling Lucky'. To the right of the search bar are links for 'Advanced Search', 'Preferences', and 'Language Tools'. At the bottom, it says 'Google.co.jp offered in: 日本語' and provides links for 'Advertising Programs', 'Business Solutions', 'About Google', and 'Go to Google.com'. The right page is the Yahoo! Japan sign-up page. It starts with a 'Hi there!' greeting and a cartoon character. It asks if the user already has an ID or mail address, with a 'Sign In' button. Below that, it says 'We'll get you set up on Yahoo! in three easy steps! Just answer a few simple questions, select an ID and password, and you'll be all set.' There's a dropdown for 'I prefer content from' set to 'Yahoo! U.S. in English'. The first step is 'Tell us about yourself...', which includes fields for 'My Name' and 'Last Name', a 'Gender' dropdown, a 'Birthday' dropdown with 'Day' and 'Year' sub-fields, an 'I live in' dropdown set to 'Japan', and a 'Postal Code' field. The second step is 'Select an ID and password', which includes a 'Yahoo! ID and Email' field, a '@ yahoo.com' dropdown, a 'Check' button, 'Password' and 'Re-type Password' fields, and a 'Password Strength' indicator. The third step is 'In case you forget your ID or password...', which includes an 'Alternate Email' field, two 'Security Question' dropdowns, and 'Your Answer' fields for each.

Google

Advanced Search
Preferences
Language Tools

Google Search I'm Feeling Lucky

Google.co.jp offered in: 日本語

Advertising Programs - Business Solutions - About Google - Go to Google.com

©2009 - Privacy

Hi there!

Already have an ID or Mail address?
Sign In

Forget your password or Yahoo! ID?

I prefer content from Yahoo! U.S. in English

1. Tell us about yourself...

My Name Last Name

Gender - Select One -

Birthday - Select Month - Day Year

I live in Japan

Postal Code

2. Select an ID and password

Yahoo! ID and Email @ yahoo.com Check

Password Password Strength

Re-type Password

3. In case you forget your ID or password...

Alternate Email

1.Security Question - Select One -

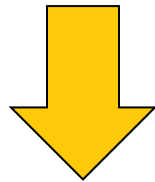
Your Answer

2.Security Question - Select One -

Your Answer

HTML Form Example

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



3. HTML Input Forms

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

The screenshot shows a Netscape 6 browser window titled "A Simple Form - Netscape 6". The address bar shows the URL "http://perl-pgm.com/php/C2/form.html". The form content is as follows:

Welcome To Our Travel Survey

Name:

May we contact you? Yes ☒ No ☐

Please check all the ways you have traveled:
☒ Walk ☐ Bicycle ☐ Car ☐ Plane

Indicate your preference for accommodations:

Any other comments?

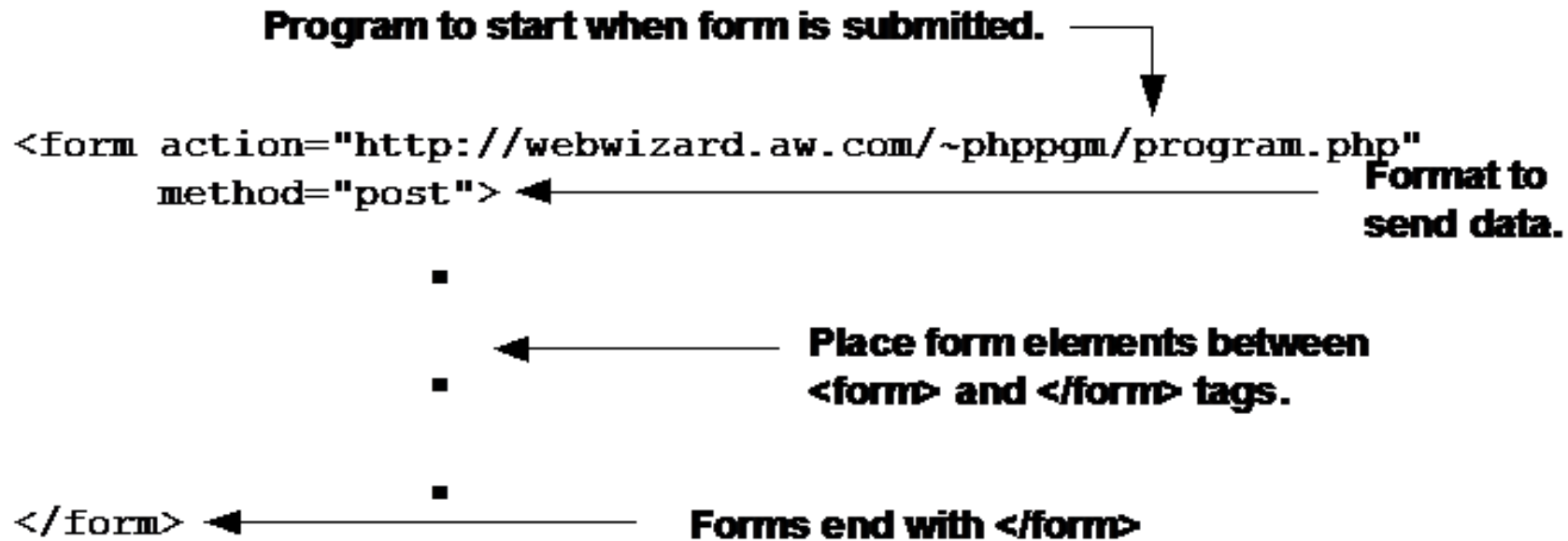
Your comments here

Labels on the right with arrows pointing to the form elements:

- Text Box (points to the Name input)
- Radio Buttons (points to the Yes/No radio buttons)
- Check Box (points to the Walk checkbox)
- Select Box (points to the accommodation dropdown menu)
- Text Area (points to the comments text area)
- Submit/Reset button (points to the Click To Submit and Erase and Restart buttons)

3.1. Starting And Ending HTML Forms

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

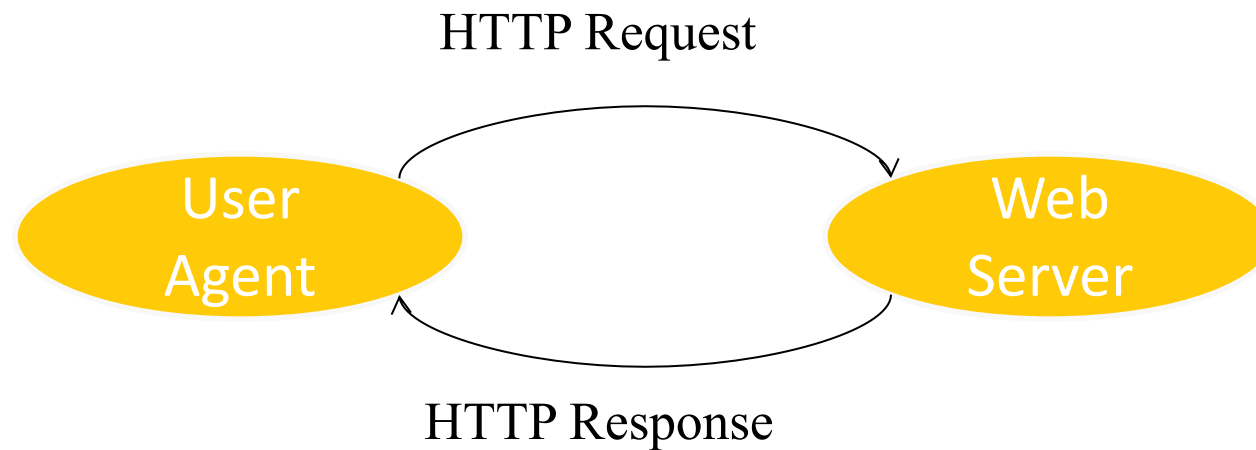


HTML Form

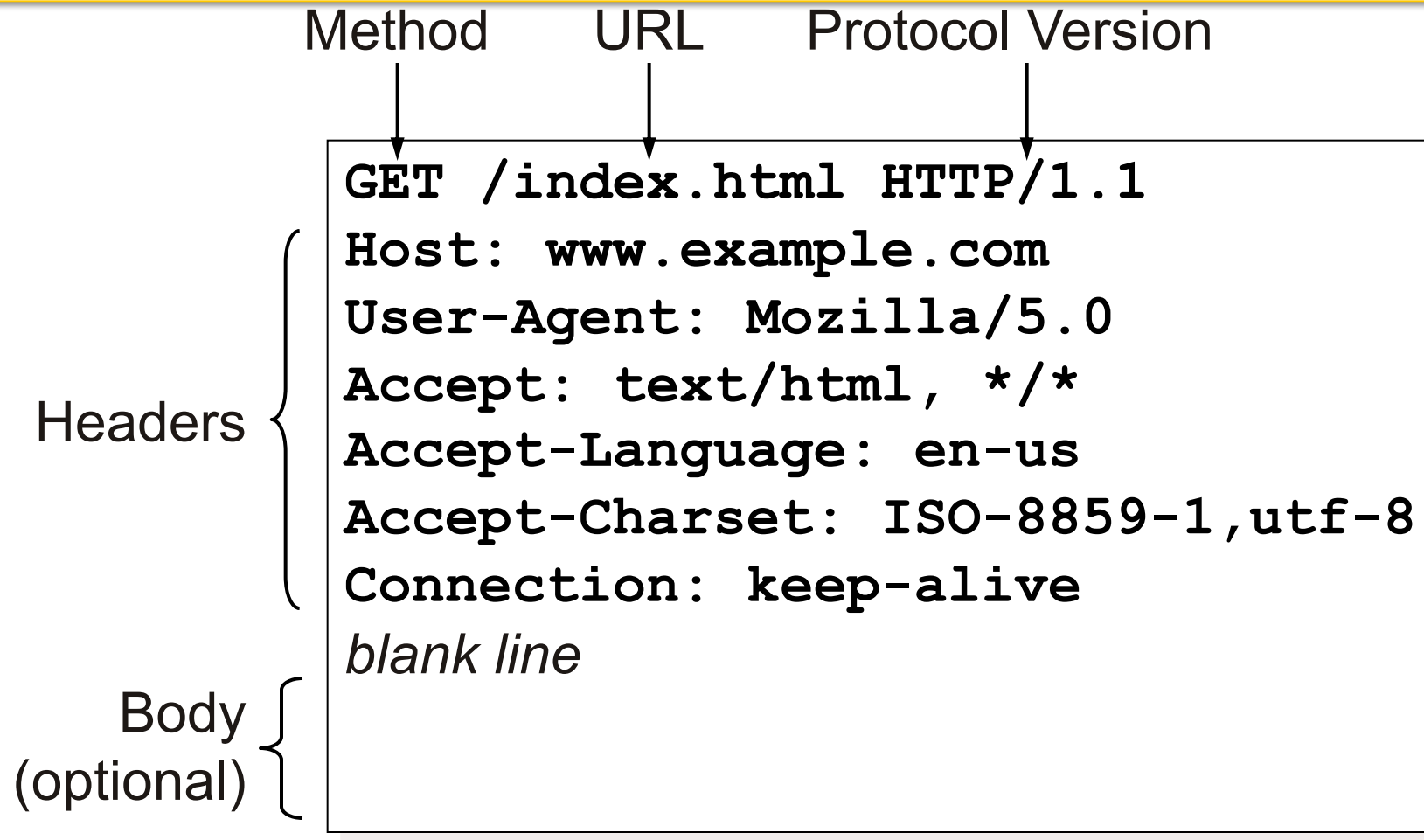
- action attribute
 - URI Reference where you want to send data
- method attribute
 - Data transfer method
 - GET
 - Send data in the query part of the URI
 - POST
 - Send data in the body of the submission

Review: Client Server Model (Web)

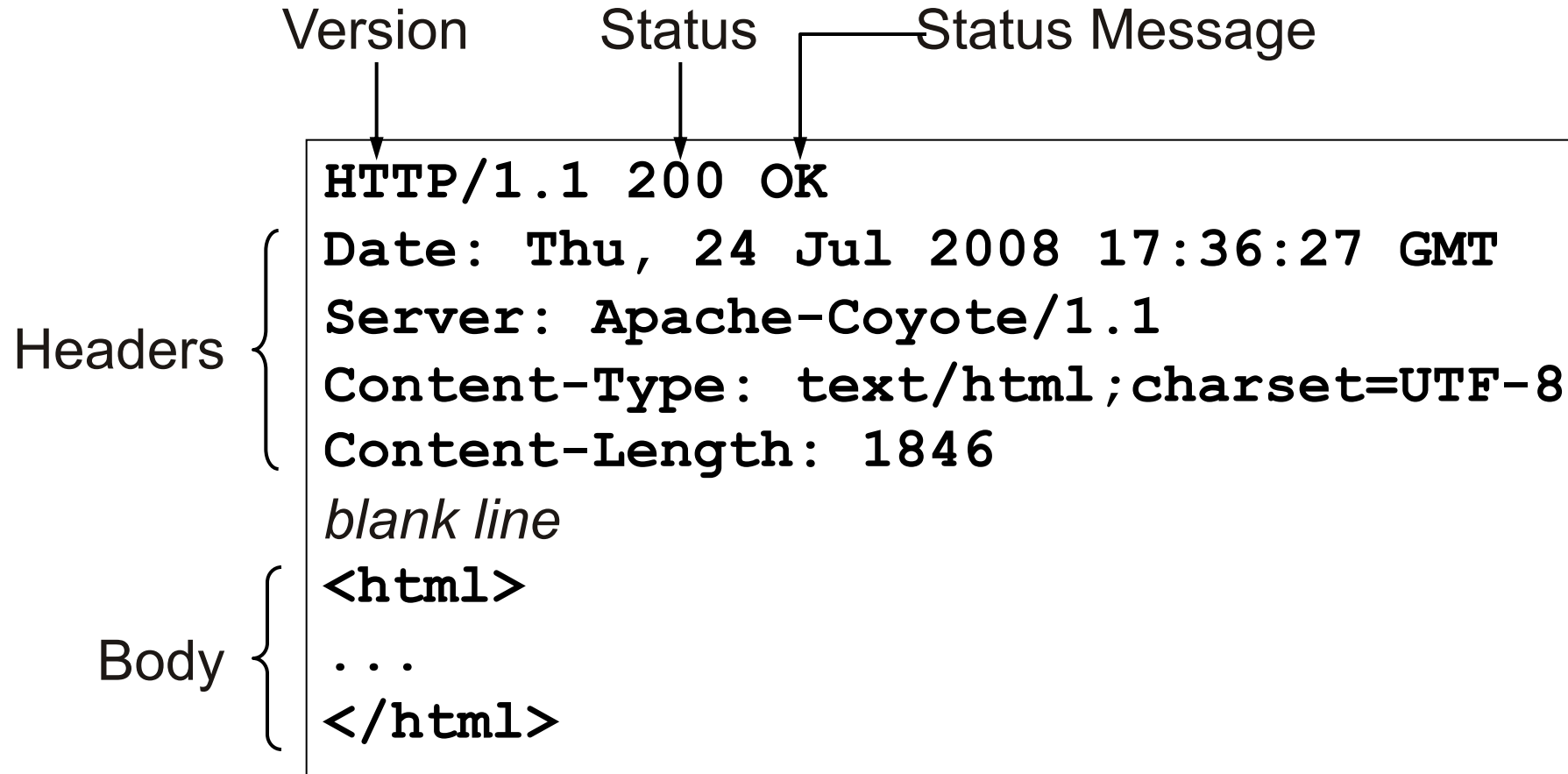
- Client: User Agent
- Server: Web server



HTTP Request



HTTP Response




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

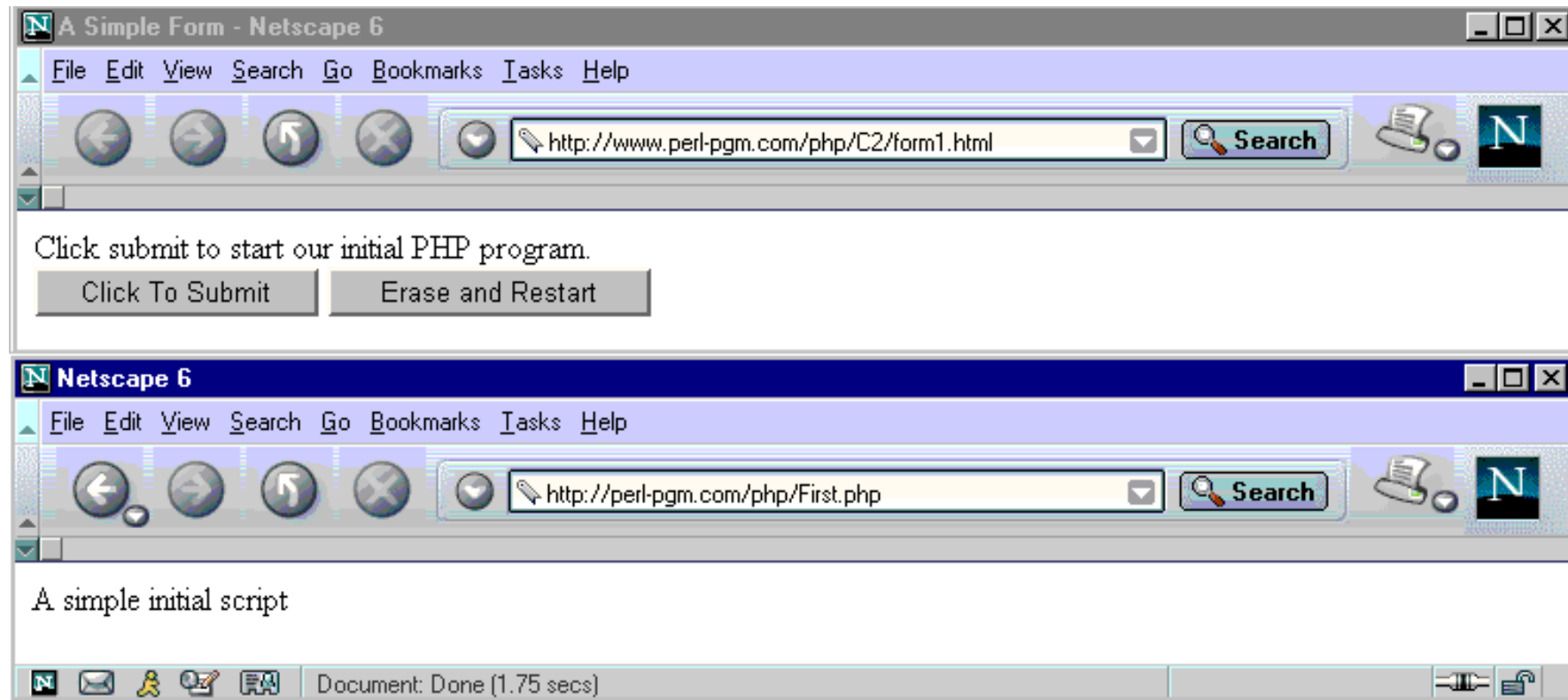


- The submit button will be labeled “Click To Submit”. The reset button will be labeled “Erase and Restart”.

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 ...



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">`

Create a text box.

The width of text box.

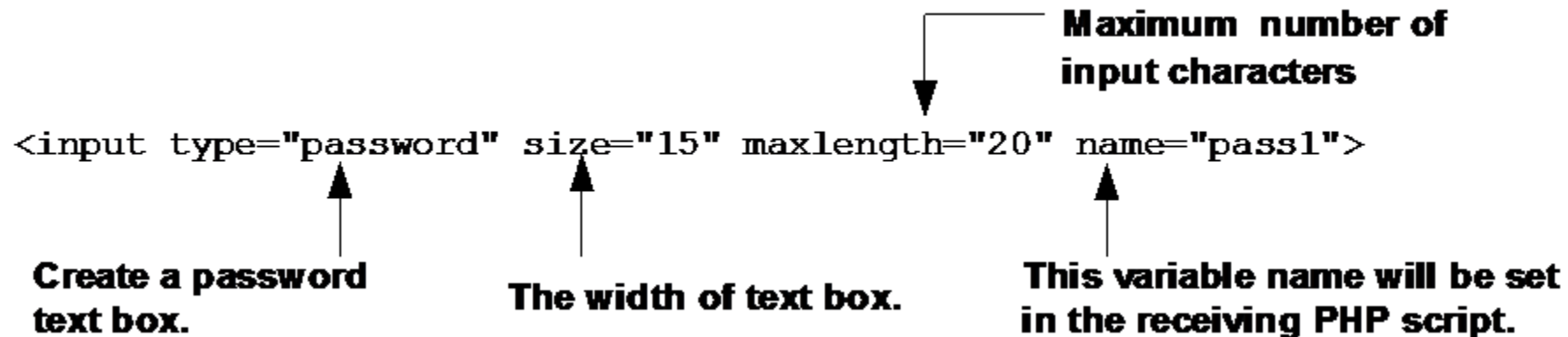
Maximum number of input characters

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).



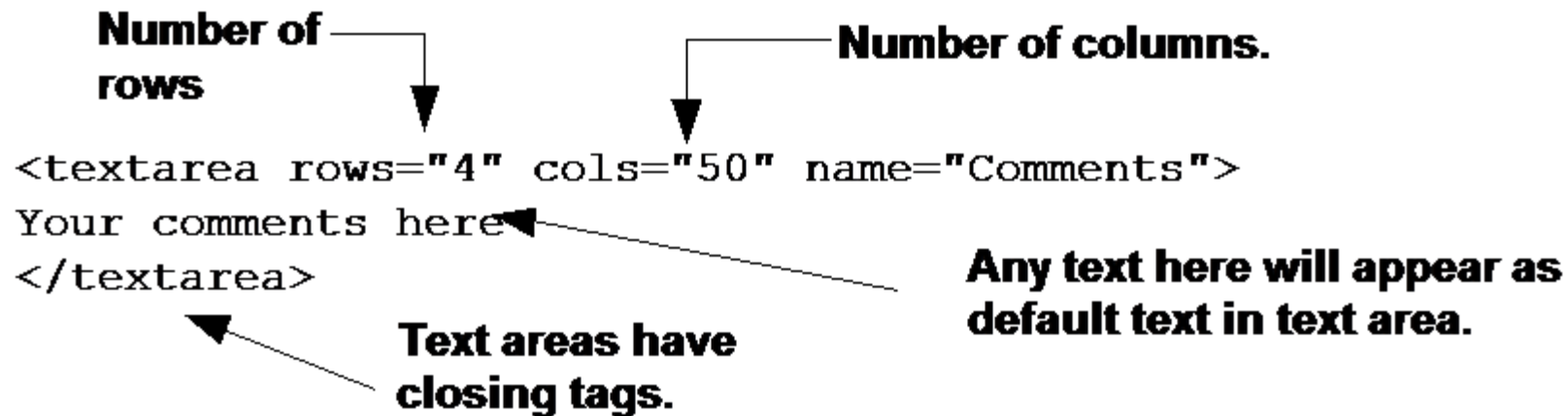
- Will be 15 characters wide accepting a maximum of 20 characters. Will set a variable named `pass1` with value of whatever the end-user enter.

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.



- 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.

The diagram shows two lines of HTML code for radio buttons. The first line is `<input type="radio" name="contact" value="Yes" checked>` and the second line is `<input type="radio" name="contact" value="No" >`. Four arrows point from text annotations to specific parts of the code: one points to `type="radio"` with the text "Create radio button."; another points to `name="contact"` with the text "Since both radio buttons have the same name, the radio buttons will operator together."; a third points to `value="Yes"` with the text "The value that will be sent to the form-processing program."; and a fourth points to `checked` with the text "This item will be pre-checked when the form is viewed."

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

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.

The diagram shows four lines of HTML code for creating checkboxes. Annotations with arrows point to specific attributes in the code:

- Annotation 1:** "This item will be pre-checked when the form is viewed." points to the `checked` attribute in the first line.
- Annotation 2:** "Create checkbox" points to the `<input type="checkbox">` part of the first line.
- Annotation 3:** "Each check box sets a different variable name when selected." points to the `name` attribute in the second, third, and fourth lines.
- Annotation 4:** "The value that will be sent to the form-processing program." points to the `value` attribute in the second, third, and fourth lines.

```
<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
```

- 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?
```

Create checkbox (points to the first `<input type="checkbox">`)

Since each checkbox element has the same name, multiple values can be set for the same variable name. (points to the `name="travel"` attribute in all four lines)

This item will be pre-checked when form is viewed. (points to the `checked` attribute in the first line)

The value that will be sent to the form-processing program. (points to the `value="None"` attribute in the fourth line)

- 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.

Variable name set in the receiving script → **Viewable window size** → **Allows end-user to select multiple items.**

```
<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>
```

↑ **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

Receiving Form Input into PHP Scripts

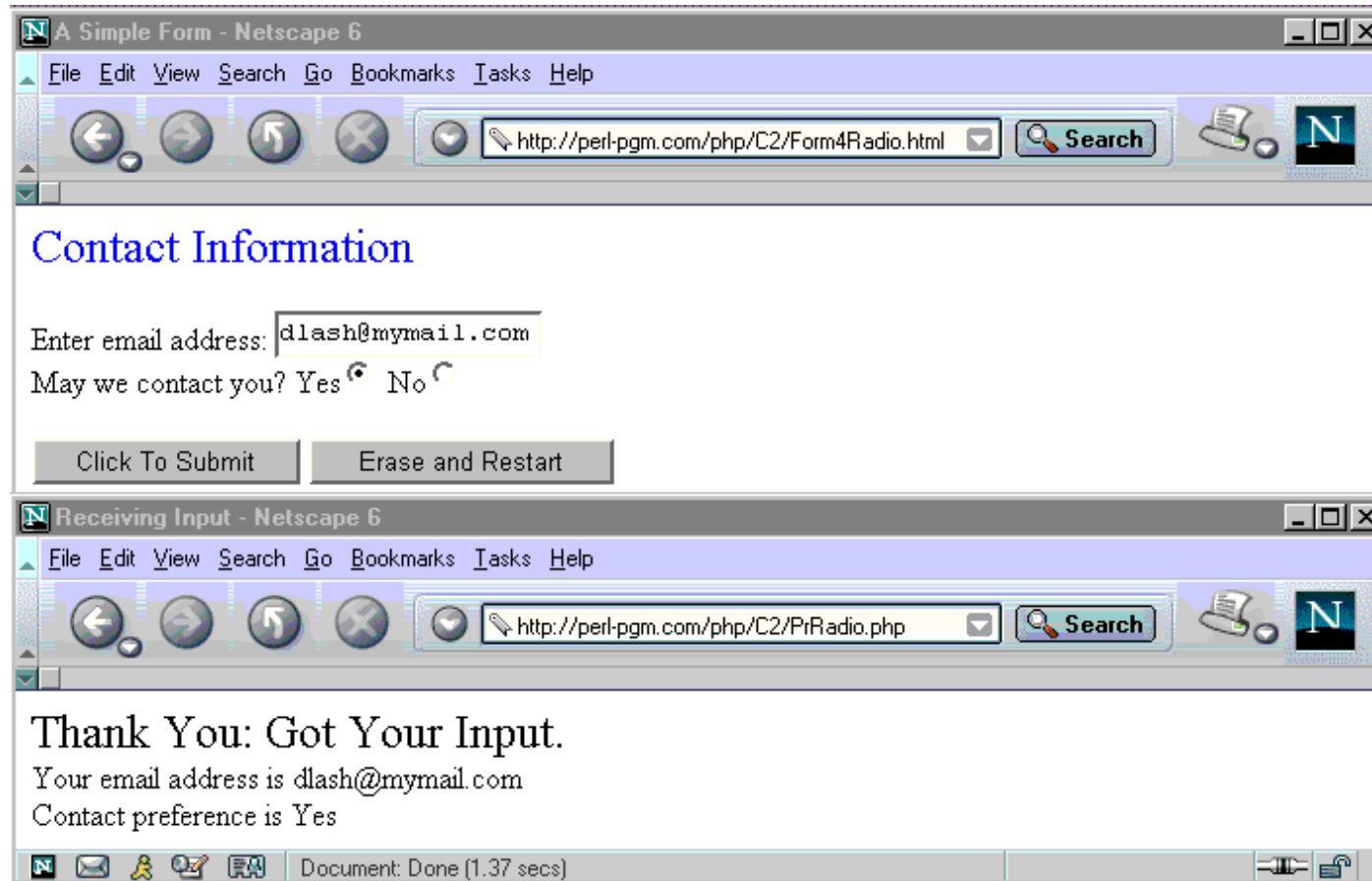
- To receive HTML form input into a PHP script:
 - Use a PHP var name that matches the variable defined in the form element's **name** argument.
- E.g., if form uses the following:
 - `<input type="radio" name="contact" value="Yes">`
- Then form-handling PHP script could use a variable called `$contact`.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

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.       print("<br> Contact preference is $contact");
8.     ?>
```

A Full Example ...



Getting input data

- To receive data, you use a special variable called `$_POST`.

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

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

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

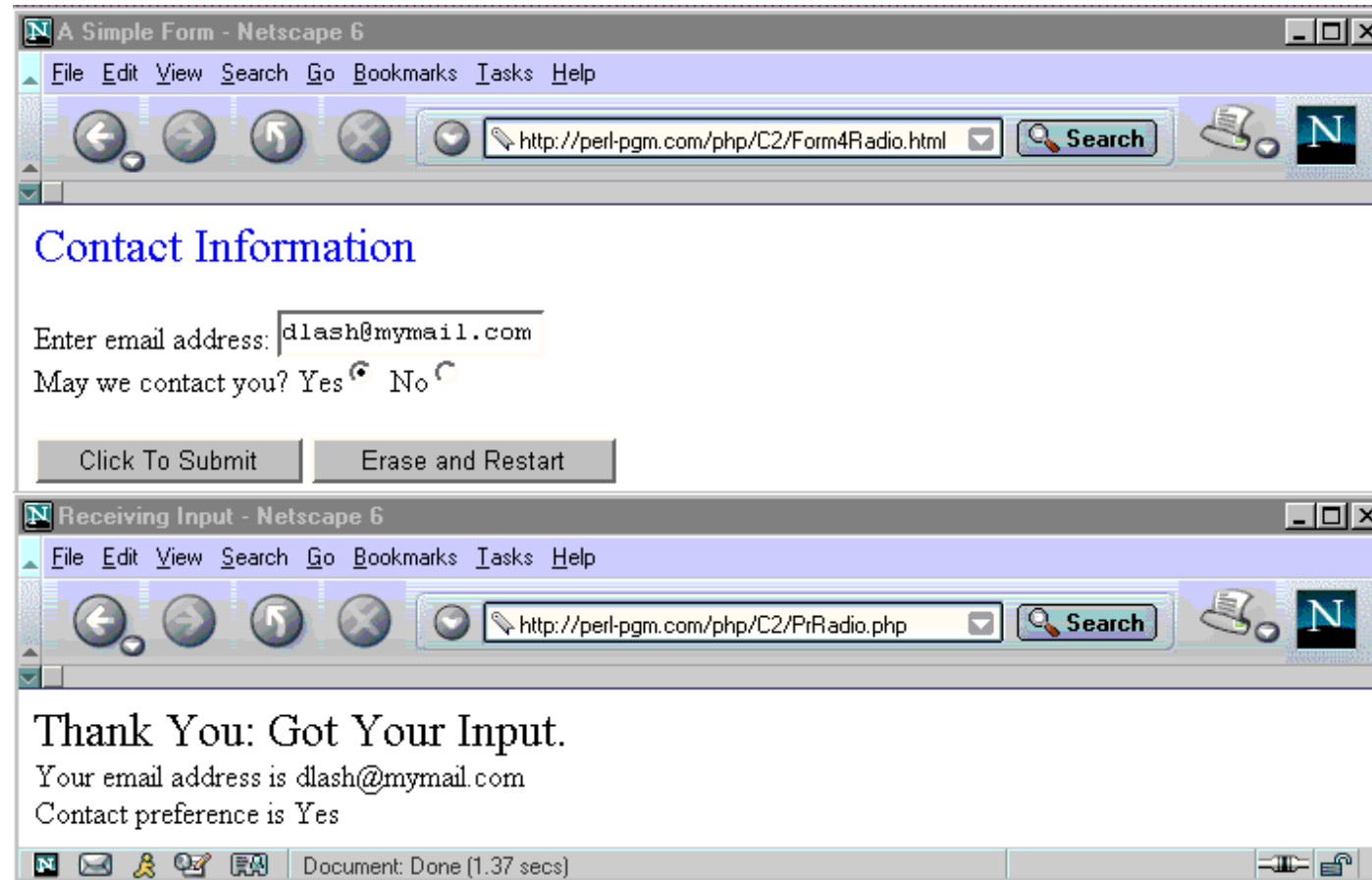
PHP variable name that you want to receive the HTML form input.

Full Example

- 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.       $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 ...



Question?

