CNT 4714: Enterprise Computing
Fall 2010

Introduction to PHP – Part 1

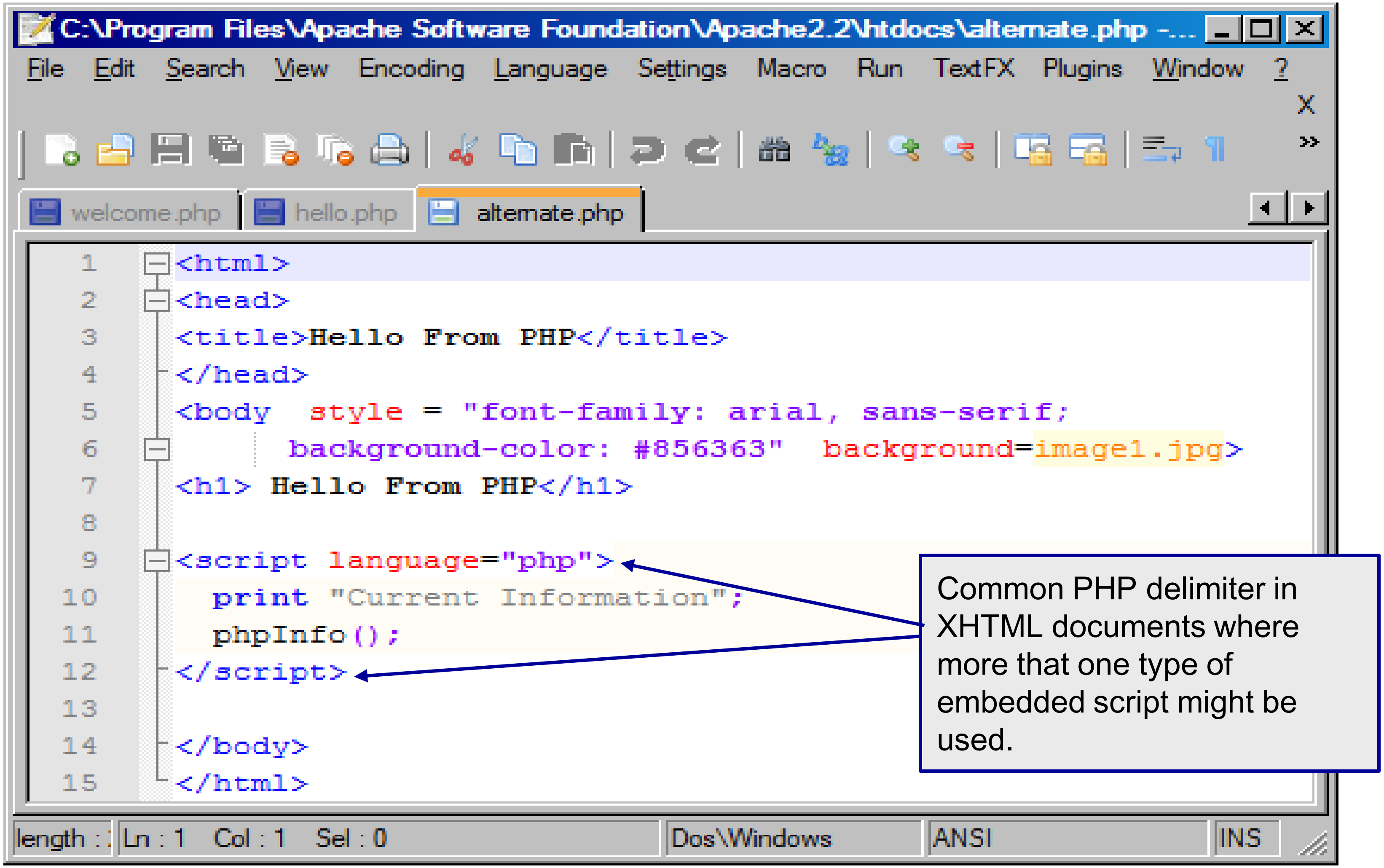
Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
http://www.cs.ucf.edu/courses/cnt4714/fall2010

Department of Electrical Engineering and Computer Science
University of Central Florida

Introduction to PHP

- We looked at a simple PHP example at the end of the set of notes that covered the installation of the Apache HTTP Server and PHP.
- PHP scripts can be created with any text editor, although Notepad++ is quite convenient for PHP scripting. I'll primarily use it in the examples.
- PHP script files should be saved with a `.php` extension.
- When PHP is embedded inside XHTML documents, as it commonly is, several different delimiters can be used. These are illustrated on the next page.

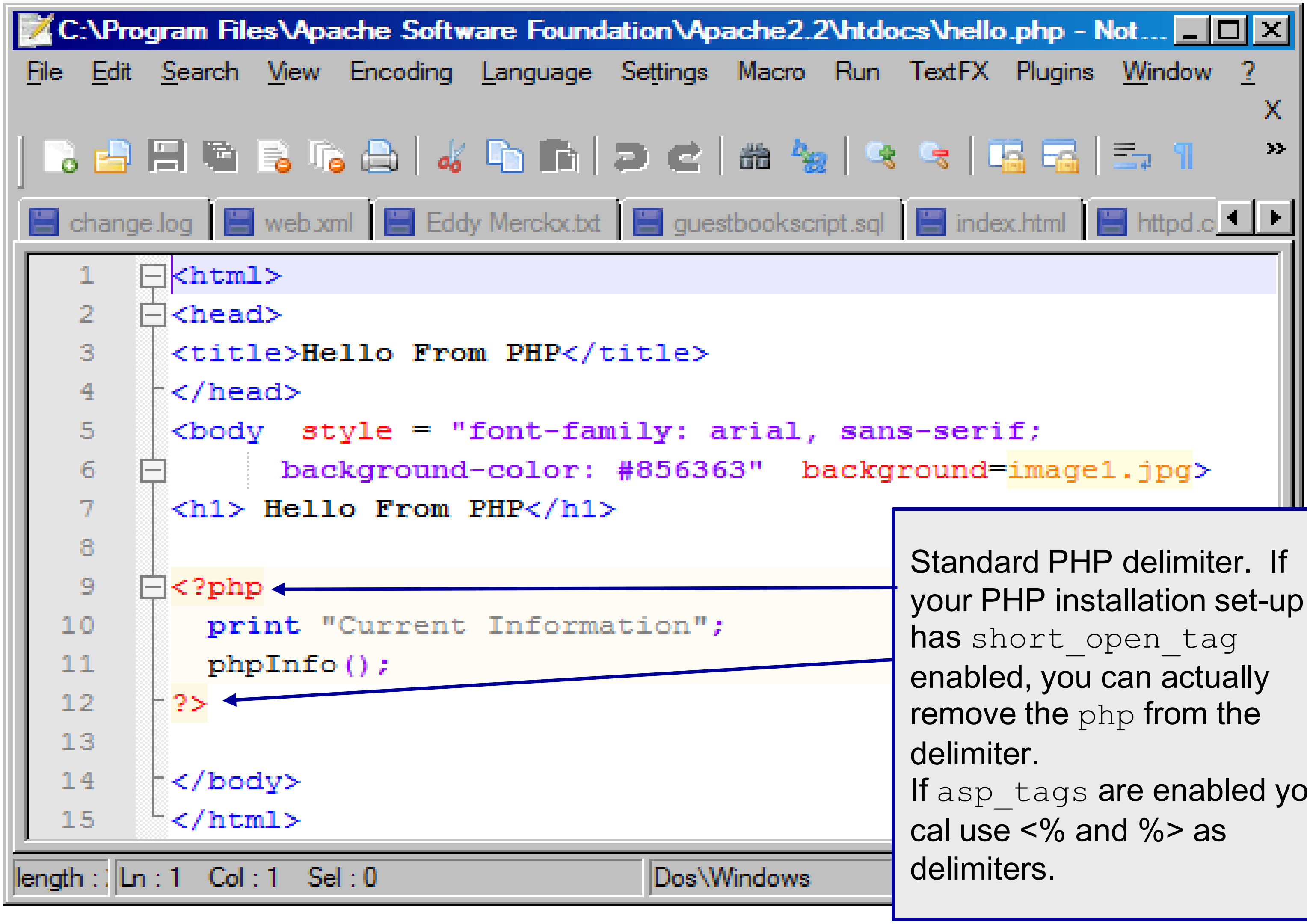
Introduction to PHP



```
<html>
<head>
<title>Hello From PHP</title>
</head>
<body  style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1.jpg>
<h1> Hello From PHP</h1>

<script language="php">
  print "Current Information";
  phpInfo();
</script>

</body>
</html>
```

Common PHP delimiter in XHTML documents where more that one type of embedded script might be used.

Introduction to PHP


C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\hello.php - Not...
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
change.log
web.xml
Eddy Merckx.txt
guestbookscript.sql
index.html
httpd.c
length :
Ln : 1 Col : 1 Sel : 0
Dos\Windows

```
<html>
<head>
<title>Hello From PHP</title>
</head>
<body  style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1.jpg>
<h1> Hello From PHP</h1>

<?php
  print "Current Information";
  phpInfo();
?>

</body>
</html>
```

Standard PHP delimiter. If your PHP installation set-up has `short_open_tag` enabled, you can actually remove the `php` from the delimiter.

If `asp_tags` are enabled you cal use <% and %> as delimiters.

Introduction to PHP

- As with any programming language, good practice in writing scripts would require comments to be included within the script.
- In-line comments in PHP are indicated with two forward slashes (//).
- Comments can appear any where in the script file and can appear in any position on any line.
- Multiple line comments are delimited with /* and */
- Most PHP implementations also allow # to delimit in-line comments.

Variables In PHP

- You can select just about any set of characters for a variable name in PHP, but they must:
  - Use a dollar sign ($) as the first character.
  - Use a letter or an underscore character (_) as the second character.
- Note: As with any programming/scripting language, good practice would suggest selecting variable names that help describe their function. For example `$counter` is more descriptive than `$c` or `$ctr`.

Variables In PHP

- That is, to print out the value of $x, write the following PHP statement:

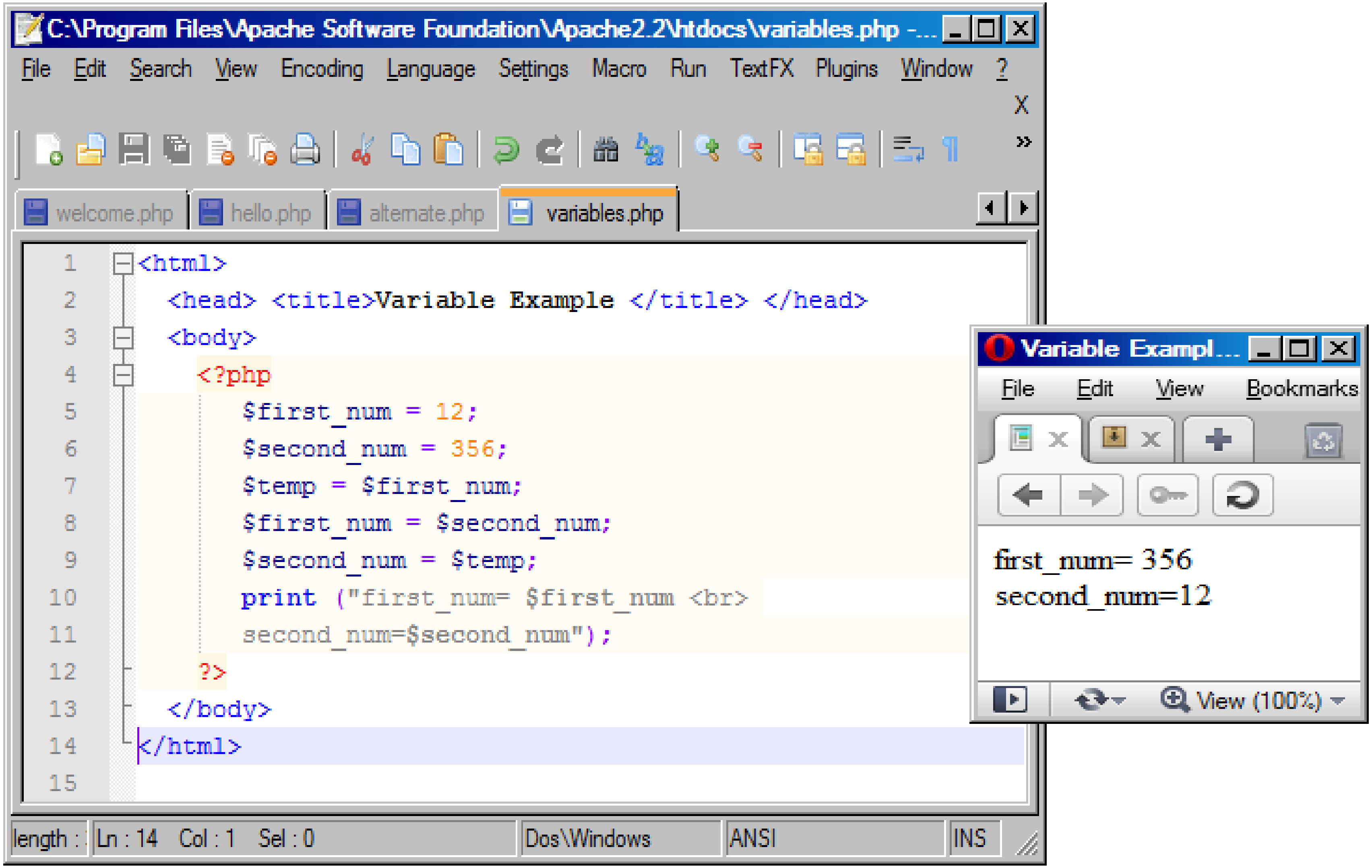
```
print ("$x");
```

- The following code will output “Candice is 26 years old”.

```
$age=26;
print (“Candice is $age years old.");
```

- The next page illustrates a full example using PHP variables.

Variables In PHP



C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\variables.php -...
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
welcome.php hello.php alternate.php variables.php
length : Ln : 14 Col : 1 Sel : 0 Dos\Windows ANSI INS
Variable Exampl...
File Edit View Bookmarks
View (100%)

```
<html>
  <head> <title>Variable Example </title> </head>
  <body>
    <?php
      $first_num = 12;
      $second_num = 356;
      $temp = $first_num;
      $first_num = $second_num;
      $second_num = $temp;
      print ("first_num= $first_num <br>
      second_num=$second_num");
    ?>
  </body>
</html>
```

```
first_num= 356
second_num=12
```

Arithmetic Operations In PHP

- PHP supports all normal arithmetic operators, with the normal semantic associated with each operator.

Operator	Effect	Example	Result
+	Addition	`$x = 2 + 2;`	`$x` is assigned 4.

-	Subtraction	`$y = 3;`
		`$y = $y - 1;`
			`$y` is assigned 2.

/	Division	`$y = 14 / 2;`	`$y` is assigned 7.

*	Multiplication	`$z = 4;`
		`$y = $z * 4;`
			`$y` is assigned 16.

%	Remainder	`$y = 14 % 3;`	`$y` is assigned 2.

- PHP supports automatic increment and decrement operations in both prefix and postfix form, i.e., -- and ++.
- Using an unassigned variable in an expression does not generate an error, the value is simply assumed to be null.

```
<?php
  $y = 3;
  $y = $y + $x + 1;
  print(“x=$x y=$y”);
?>
```

The output is: `x=y=4`

String Variables In PHP

- PHP supports character string variables and this is a widely used aspect of PHP in handling form data.
- Be careful in PHP not to mix numeric and string types together in an expression.
- 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”);
?>
```

String Variables In PHP

- The string concatenation operator in PHP is the period as shown below:

```
<?php
   $firstname = “Megan”;
   $lastname = “Fox”
   $fullname = $firstname . $lastname;
   print(“Full name = $fullname”);
?>
```

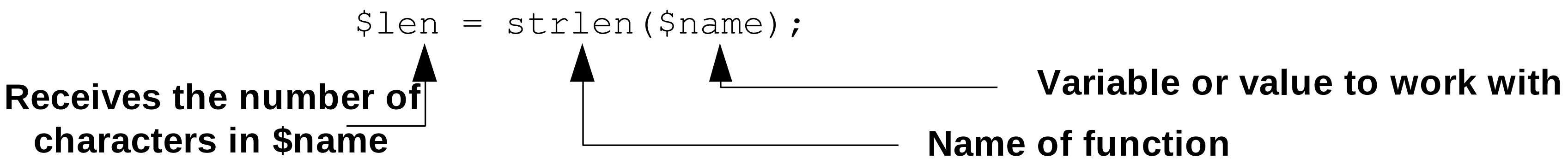
The output of this script would be: `Fullname=MeganFox`

You can also use double quotation marks to create concatenation directly. Using the above example you could do the following: $fullname2 = “$firstname $lastname”; This would have the same effect as: $fullname2 = $firstname . $lastname;

String Variables In PHP

- PHP supports a large variety of string handling functions. A few of the more commonly used ones are illustrated on the next few pages.

- 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

String Variables In PHP

`strlen()` function:

- This function returns the number of characters in the string argument to the function. Consider the following script:

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

This PHP script would output “`Length=8`”.

String Variables In PHP

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 = “       Megan  Fox  ";
  $name = trim($in_name);
  print ("name=$name$name");
?>
```

This PHP script would output “name=Megan FoxMegan Fox”.

String Variables In PHP

strtolower() and strtoupper functions:

- These functions return the argument string in all uppercase or all lowercase letters, respectively. For example, consider the following script:

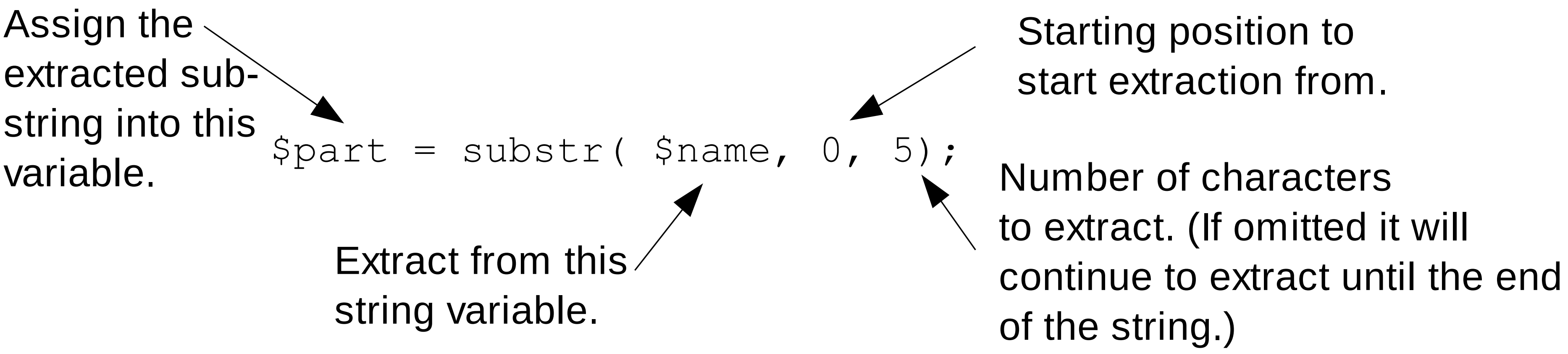
```
<?php
  $inquote = “Now Is The Time";
  $lower = strtolower($inquote);
  $upper = strtoupper($inquote);
  print(“upper=$upper lower=$lower”);
?>
```

This PHP script would output “`upper=NOW IS THE TIME lower = now is the time`”

String Variables In PHP

`substr()` function:

- This function enables a PHP script to extract a portion of the characters in a string variable. The general syntax is:



```
$part = substr( $name, 0, 5);
```

Assign the extracted sub-string into this variable.

Starting position to start extraction from.

Extract from this string variable.

Number of characters to extract. (If omitted it will continue to extract until the end of the string.)

String Variables In PHP

`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");
?>
```

String Variables In PHP

`substr()` function:

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

```
<?php
  $date = "12/25/2010";
  $year = substr($date, 6);
  print ("Year=$year");
?>
```

- The above script segment would output “`Year=2010`”.

Controlling Script Flow In PHP

- PHP contains the normal control statements that handle decision making and iteration within a script.
- Normal logical operators are all supported with their standard semantic.
- As with many modern programming and scripting languages remember to use == in a logical comparison operation and not =. The single equal sign is an assignment operator and as such is always true. No syntax error is generated.
- The table on the following page illustrates the common logical operators in PHP.

Controlling Script Flow In PHP

Test Operator	Effect	Example	Result

`==`	Equal to	`if ($x == 6){`
		`    $x = $y + 1;`
		`    $y = $x + 1;`
		`}`
			Run the second and third statements if the value of `$x` *is equal to* `6`.

`!=`	Not equal to	`if ($x != $y) {`
		`    $x = 5 + 1;`
		`}`
			Run the second statement if the value of `$x` *is not equal to* the value of `$y`.

`<`	Less than	`if ($x < 100) {`
		`    $y = 5;`
		`}`
			Run the second statement if the value of `$x` *is less than* `100`.

`>`	Greater than	`if ($x > 51) {`
		`    print "OK";`
		`}`
			Run the second statement if the value of `$x` *is greater than* `51`.

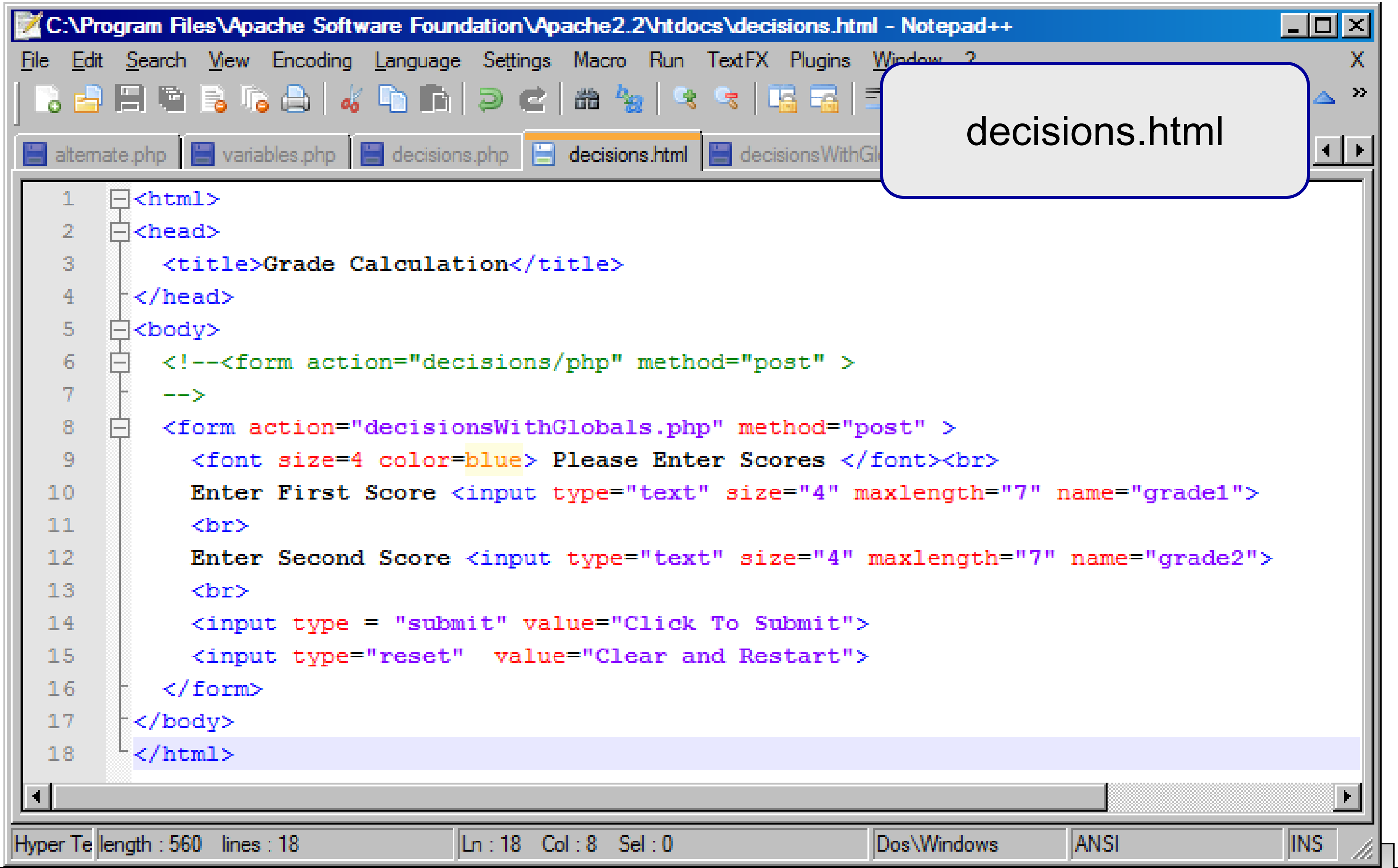
`>=`	Greater than or equal to	`if (16 >= $x) {`
		`    print "x=$x" ;`
		`}`
			Run the second statement if `16` *is greater than or equal to* the value of `$x`.

`<=`	Less than or equal to	`if ($x <= $y) {`
		`    print "y=$y" ;`
		`    print "x=$x" ;`
		`}`
			Run the second and third statements if the value of `$x` *is less than or equal to* the value of `$y`.

Controlling Script Flow In PHP

- The following example uses an input form (XHTML) and two values are extracted from the form (grade1 and grade2), passed to a PHP script which determines the average score, the maximum score and assigns a grade to the average for the student’s scores.
- We’ll get much more into forms and form handling in PHP later, but this simple example will illustrate several of the common threads that appear in form handling in PHP (and server side scripting in general).

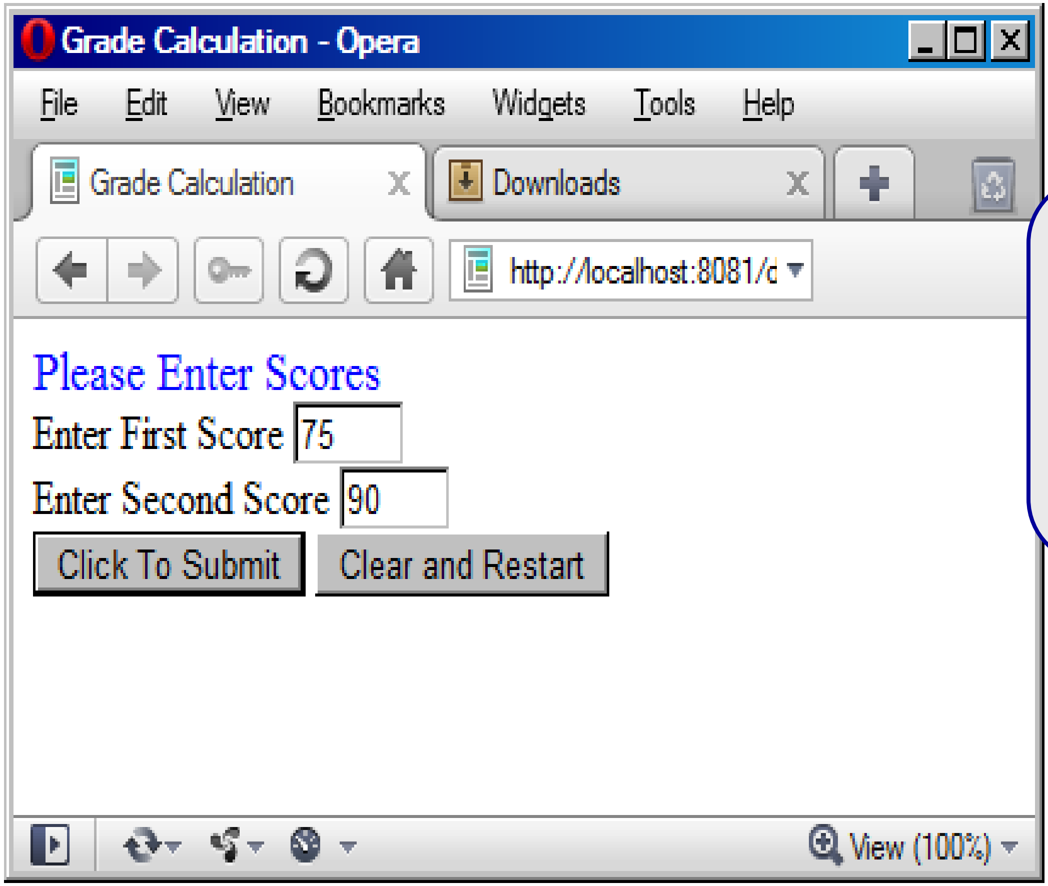
Controlling Script Flow In PHP


C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\decisions.html - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
alternate.php
variables.php
decisions.php
decisions.html
decisionsWithGl
Hyper Te
length : 560 lines : 18
Ln : 18 Col : 8 Sel : 0
Dos\Windows
ANSI
INS

```
<html>
<head>
  <title>Grade Calculation</title>
</head>
<body>
  <!--<form action="decisions/php" method="post" >
  -->
  <form action="decisionsWithGlobals.php" method="post" >
    <font size=4 color=blue> Please Enter Scores </font><br>
    Enter First Score <input type="text" size="4" maxlength="7" name="grade1">
    <br>
    Enter Second Score <input type="text" size="4" maxlength="7" name="grade2">
    <br>
    <input type = "submit" value="Click To Submit">
    <input type="reset"  value="Clear and Restart">
  </form>
</body>
</html>
```

decisions.html

Controlling Script Flow In PHP


Grade Calculation - Opera
File Edit View Bookmarks Widgets Tools Help
Grade Calculation
Downloads
http://localhost:8081/c
View (100%)

Please Enter Scores

Enter First Score 75

Enter Second Score 90

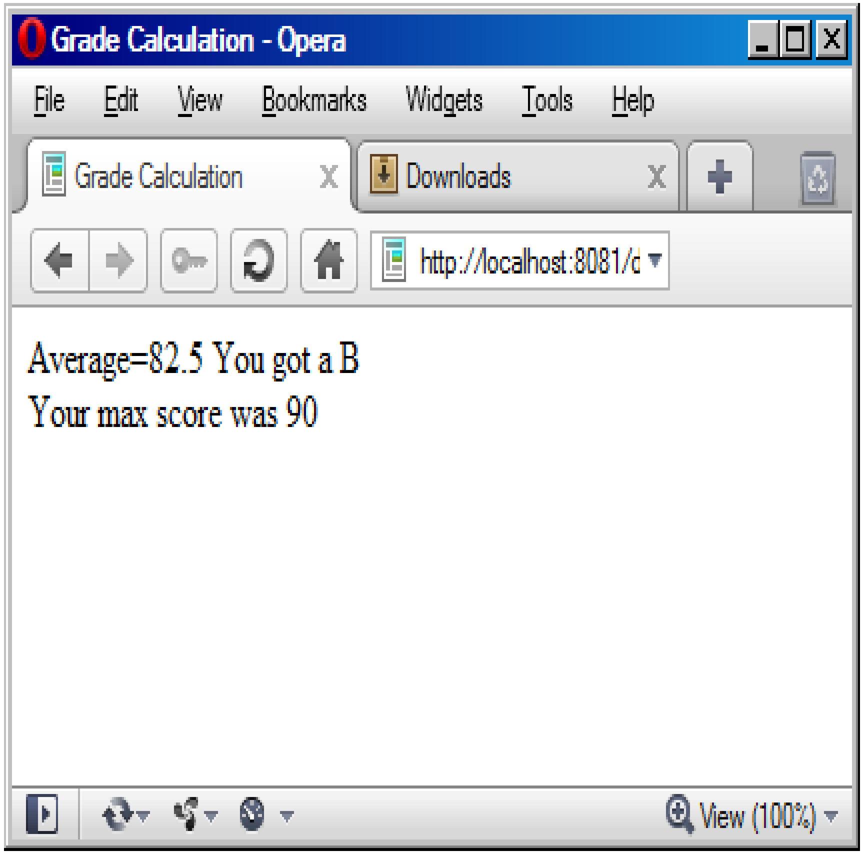
Click To Submit

Clear and Restart

Executing `decisions.html`

User enters two scores, clicks submit button.

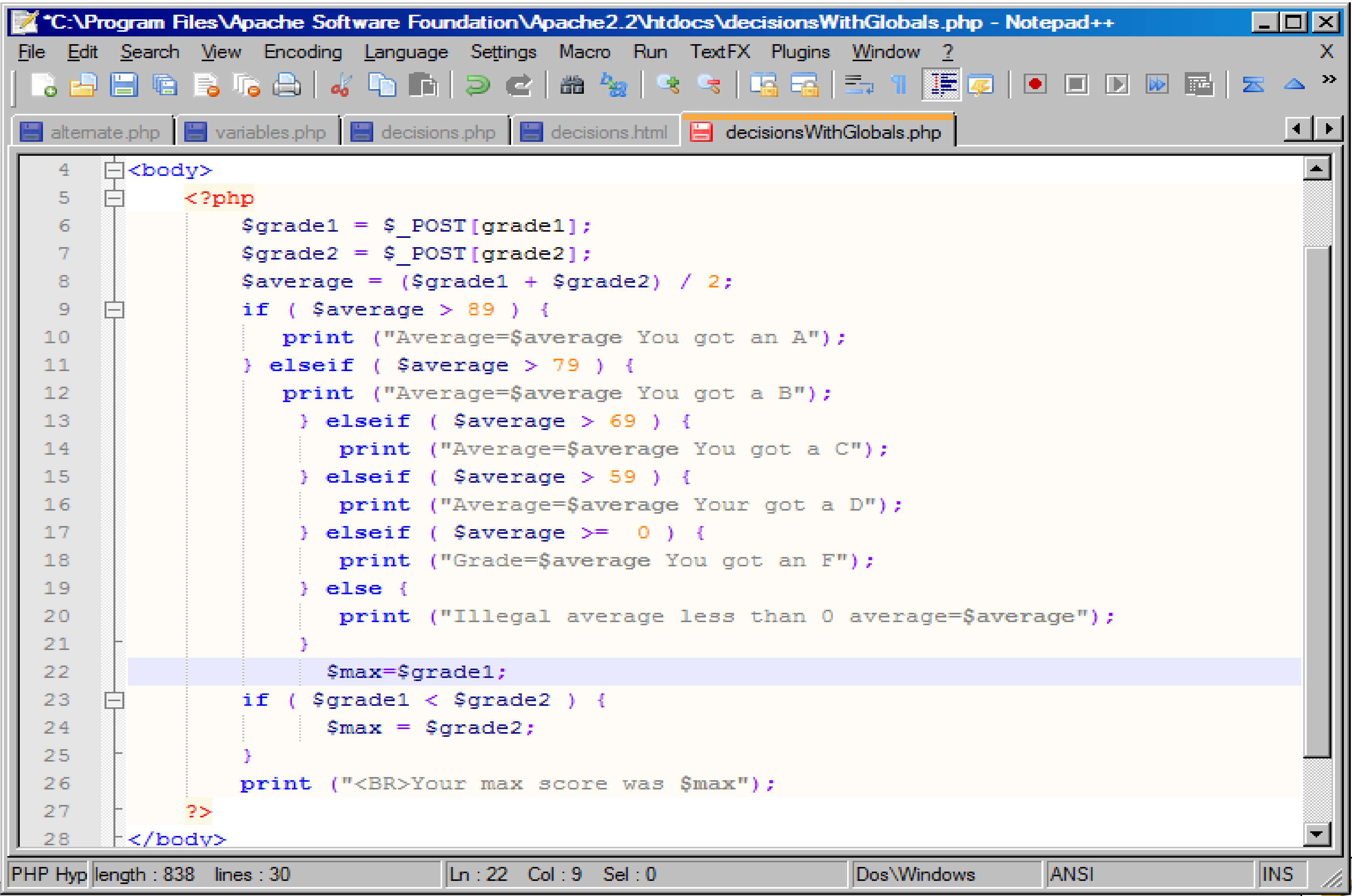
Controlling Script Flow In PHP


Grade Calculation - Opera
File Edit View Bookmarks Widgets Tools Help
Grade Calculation
Downloads
http://localhost:8081/c
View (100%)

Average=82.5 You got a B
Your max score was 90

Clicking the submit button triggers the action of the form and invokes the script `decisions.php` The script generates this page. The PHP script is shown on the next page.

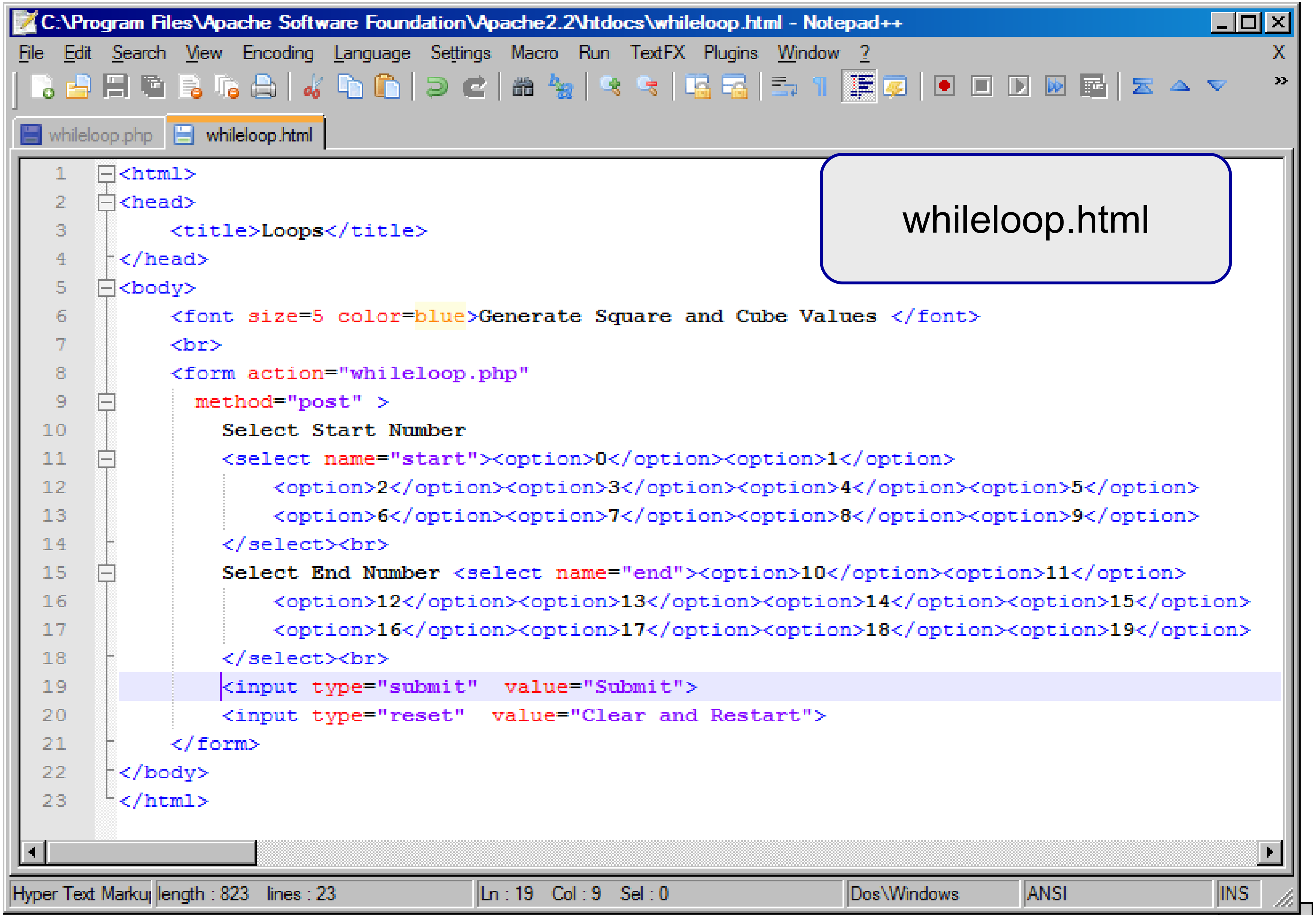
Controlling Script Flow In PHP


*C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\decisionsWithGlobals.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
alternate.php
variables.php
decisions.php
decisions.html
decisionsWithGlobals.php
PHP Hyp
length : 838 lines : 30
Ln : 22 Col : 9 Sel : 0
Dos\Windows
ANSI
INS

```
<body>
    <?php
        $grade1 = $_POST[grade1];
        $grade2 = $_POST[grade2];
        $average = ($grade1 + $grade2) / 2;
        if ( $average > 89 ) {
           print ("Average=$average You got an A");
        } elseif ( $average > 79 ) {
           print ("Average=$average You got a B");
            } elseif ( $average > 69 ) {
               print ("Average=$average You got a C");
            } elseif ( $average > 59 ) {
               print ("Average=$average Your got a D");
            } elseif ( $average >=  0 ) {
               print ("Grade=$average You got an F");
            } else {
               print ("Illegal average less than 0 average=$average");
            }
              $max=$grade1;
        if ( $grade1 < $grade2 ) {
              $max = $grade2;
        }
        print ("<BR>Your max score was $max");
    ?>
</body>
```

Controlling Script Flow In PHP

- PHP supports three types of iterative constructs:
  - the `for` loop
  - the `while` loop
  - and the `foreach` loop.
- The `for` and `while` loops act as you would expect given your knowledge of other programming languages. The `foreach` loop applies specifically to arrays in PHP. We'll look at the `foreach` loop later.
- The example on the next couple of pages illustrates a while loop. Again, I've used a form to extract user input. This time the user input sets the lower and upper limit on the loop.

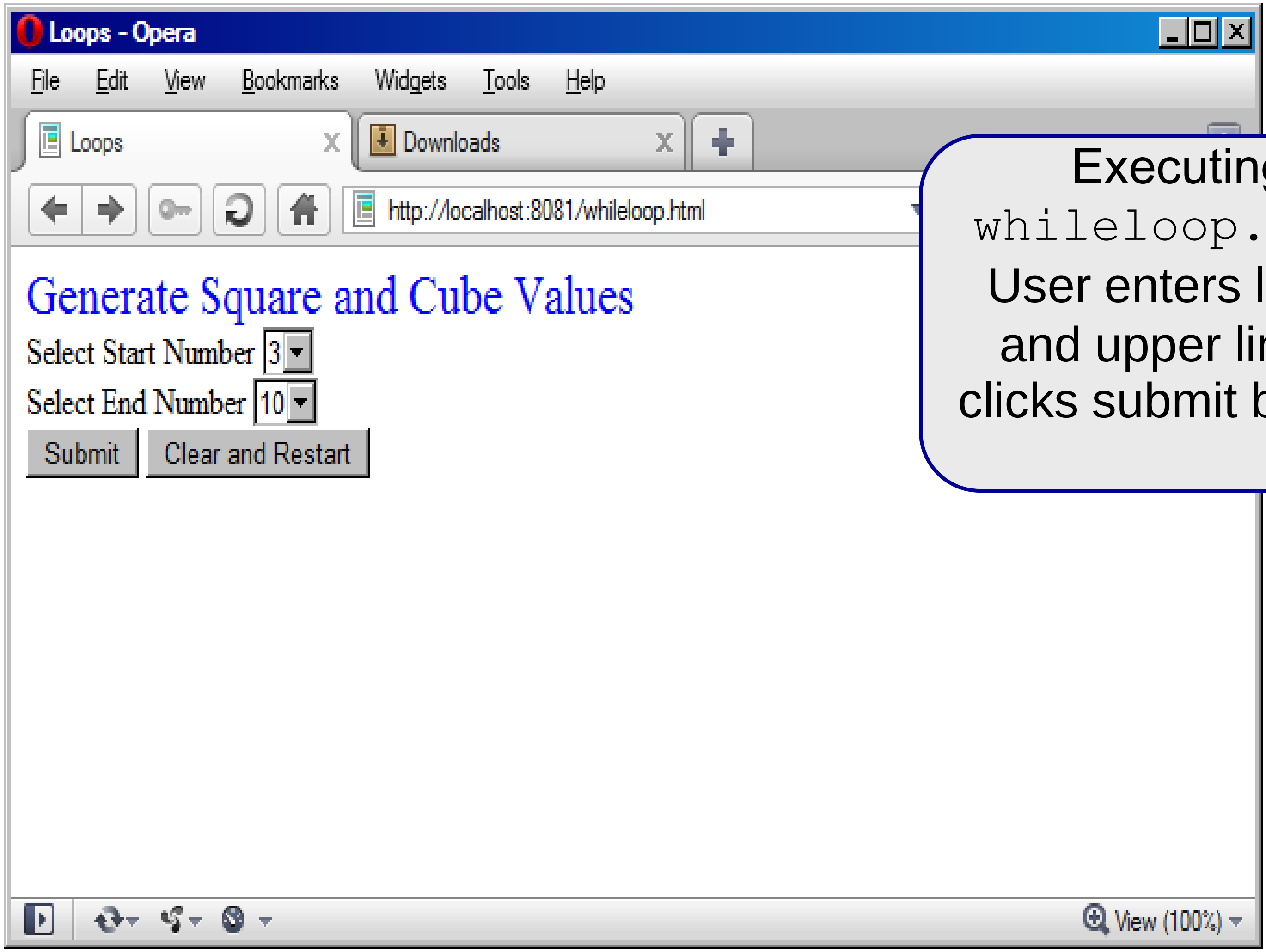

C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\whileloop.html - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
whileloop.php
whileloop.html

whileloop.html

```
<html>
<head>
    <title>Loops</title>
</head>
<body>
    <font size=5 color=blue>Generate Square and Cube Values </font>
    <br>
    <form action="whileloop.php"
      method="post" >
        Select Start Number
        <select name="start"><option>0</option><option>1</option>
            <option>2</option><option>3</option><option>4</option><option>5</option>
            <option>6</option><option>7</option><option>8</option><option>9</option>
        </select><br>
        Select End Number <select name="end"><option>10</option><option>11</option>
            <option>12</option><option>13</option><option>14</option><option>15</option>
            <option>16</option><option>17</option><option>18</option><option>19</option>
        </select><br>
        <input type="submit"  value="Submit">
        <input type="reset"  value="Clear and Restart">
    </form>
</body>
</html>
```

Hyper Text Marku length : 823 lines : 23 Ln : 19 Col : 9 Sel : 0 Dos\Windows ANSI INS

Controlling Script Flow In PHP


Loops - Opera
File Edit View Bookmarks Widgets Tools Help
Loops
Downloads
http://localhost:8081/whileloop.html
View (100%)

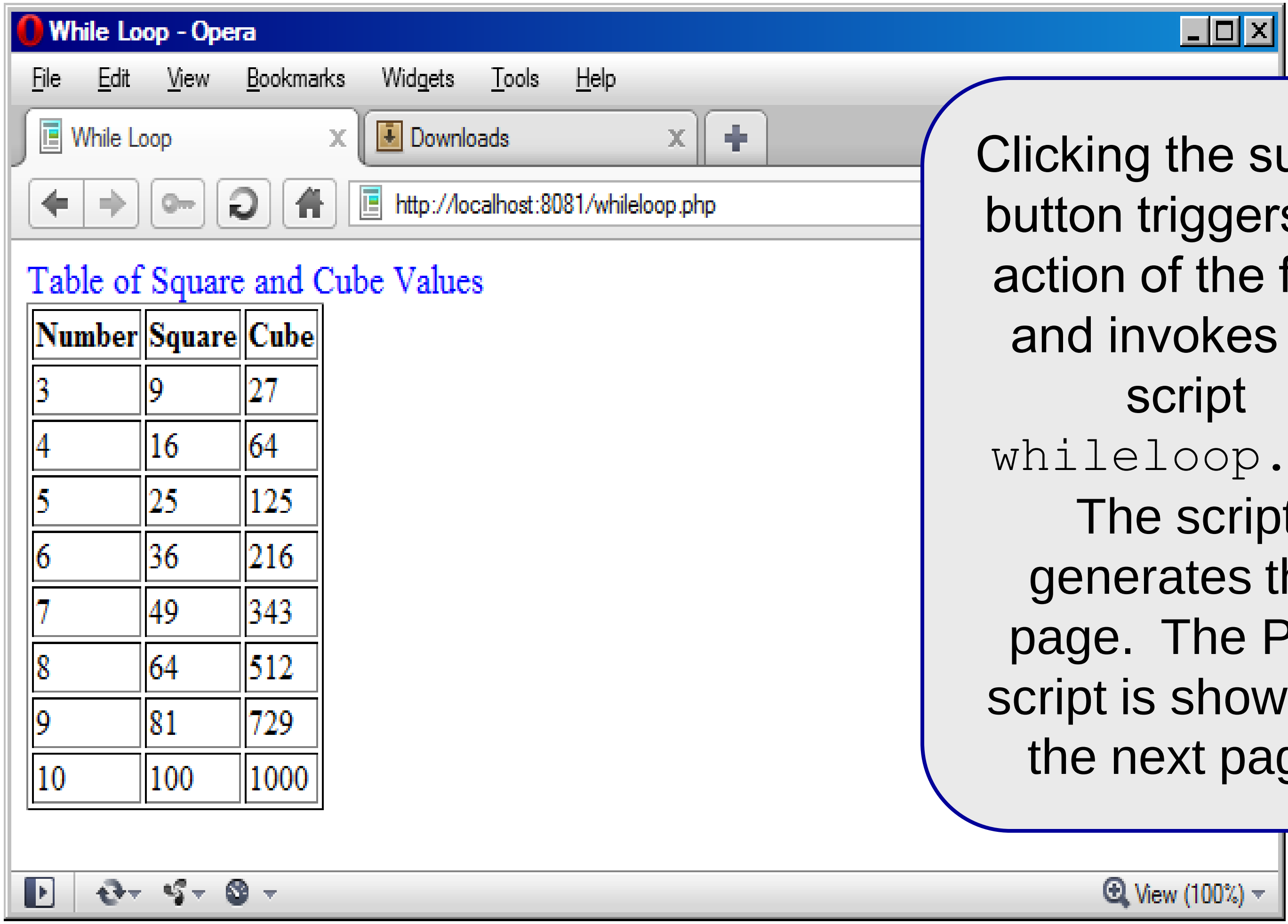
Generate Square and Cube Values

Select Start Number 3

Select End Number 10

Executing `whileloop.html` User enters lower and upper limits, clicks submit button.

Controlling Script Flow In PHP

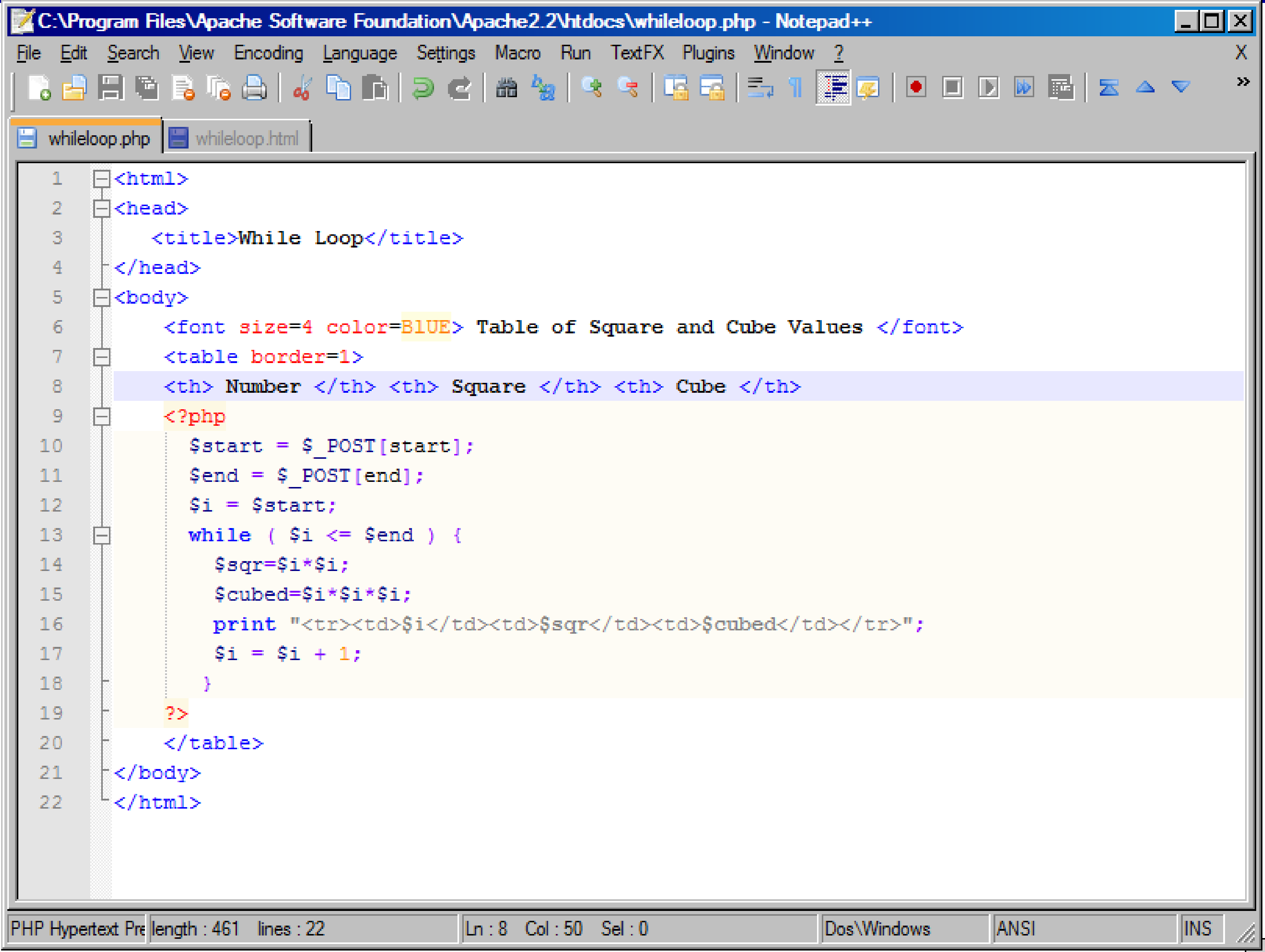


While Loop - Opera
File Edit View Bookmarks Widgets Tools Help
While Loop
Downloads
http://localhost:8081/whileloop.php
View (100%)

Table of Square and Cube Values

Number	Square	Cube
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Clicking the submit button triggers the action of the form and invokes the script `whileloop.php` The script generates this page. The PHP script is shown on the next page.


C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\whileloop.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
whileloop.php
whileloop.html

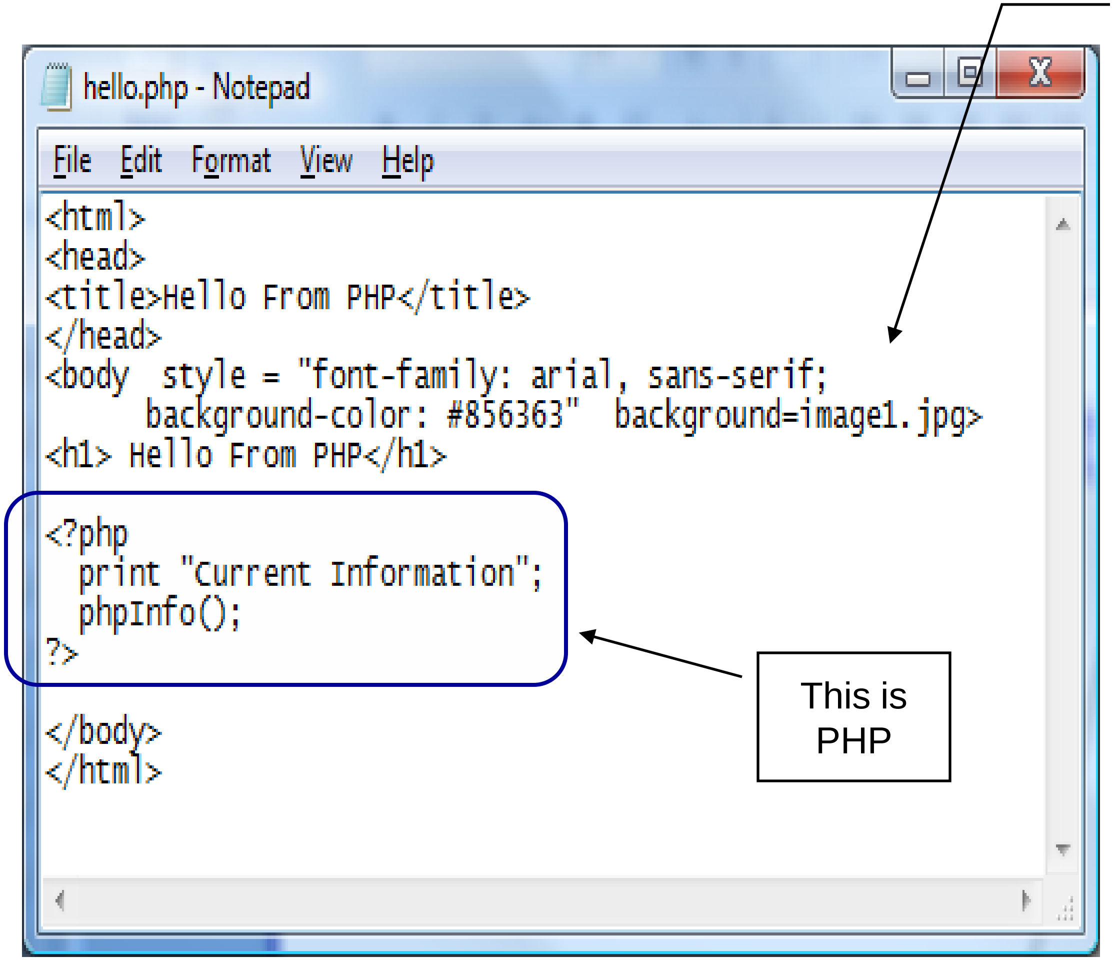
```
<html>
<head>
  <title>While Loop</title>
</head>
<body>
  <font size=4 color=BLUE> Table of Square and Cube Values </font>
  <table border=1>
  <th> Number </th> <th> Square </th> <th> Cube </th>
  <?php
   $start = $_POST[start];
   $end = $_POST[end];
   $i = $start;
   while ( $i <= $end ) {
    $sqr=$i*$i;
    $cubed=$i*$i*$i;
    print "<tr><td>$i</td><td>$sqr</td><td>$cubed</td></tr>";
    $i = $i + 1;
   }
  ?>
  </table>
</body>
</html>
```

PHP Hypertext Pre length : 461 lines : 22 Ln : 8 Col : 50 Sel : 0 Dos\Windows ANSI INS

Using PHP With REGISTER_GLOBALS OFF

- Since PHP 4.2.0, PHP is shipped with the REGISTER_GLOBALS configuration variable set to OFF. Prior to version 4.2.0 this variable was set to ON, but represented a fairly large security concern, so since that time the default setting is OFF. While this setting can be overridden by local system administrators, it is wise not to do so.
- When PHP is configured with REGISTER_GLOBALS set to OFF, you need an extra step to receive input from forms, cookies, or session variables.
- You can tell your PHP site's status of REGISTER_GLOBALS by running the `phpInfo()` function that was shown in the `hello.php` script in the setting up PHP section of notes (repeated here)…

A PHP Test Example

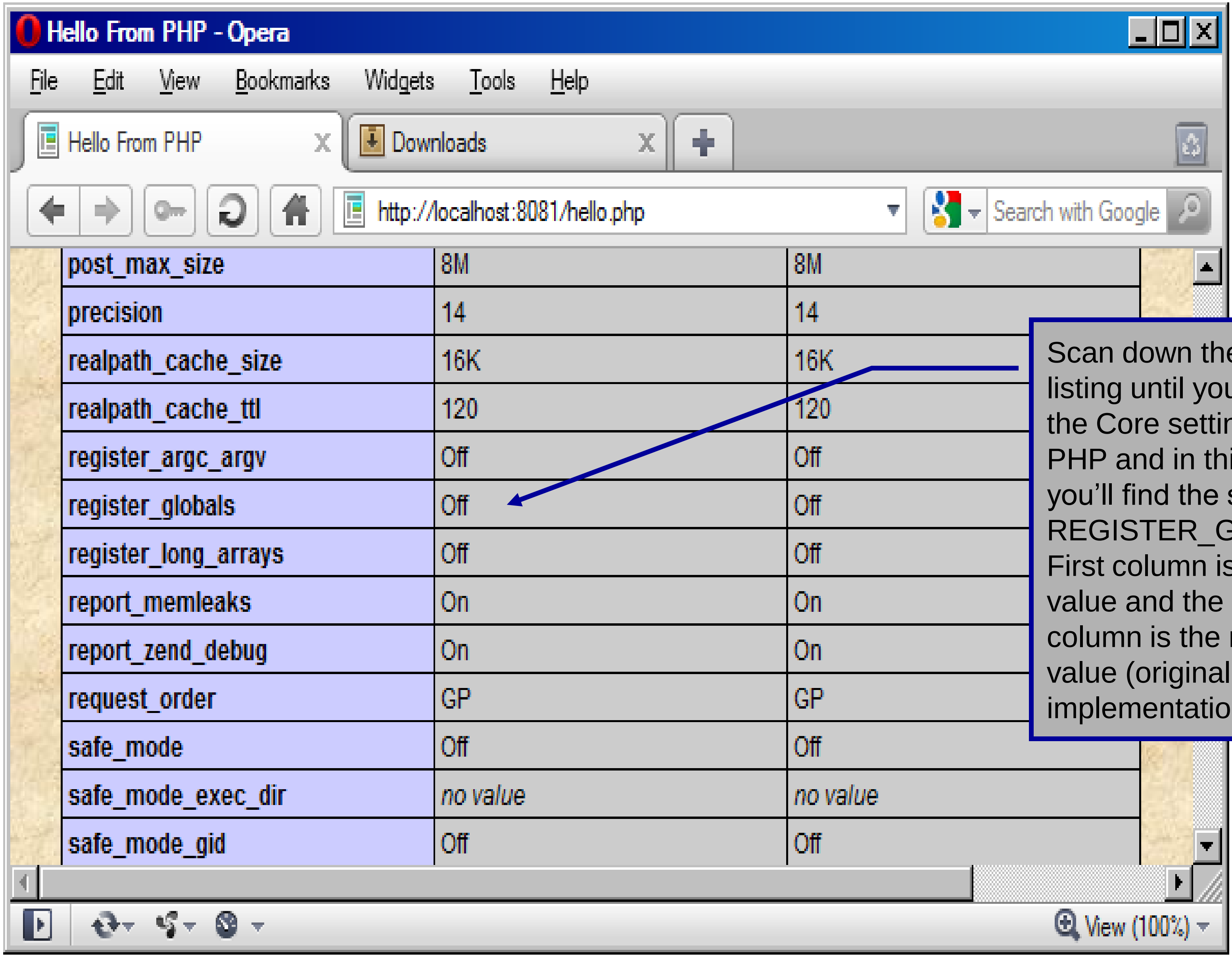


```
<html>
<head>
<title>Hello From PHP</title>
</head>
<body  style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1.jpg>
<h1> Hello From PHP</h1>

<?php
  print "Current Information";
  phpInfo();
?>

</body>
</html>
```

This is PHP

Create this file named `hello.php` and save it to the `htdocs` folder in the Apache server.

Then start your browser and enter the URL: http://localhost:8081/hello.php and you should see output similar to that shown on the next slide.



post_max_size	8M	8M
precision	14	14
realpath_cache_size	16K	16K
realpath_cache_ttl	120	120
register_argc_argv	Off	Off
register_globals	Off	Off
register_long_arrays	Off	Off
report_memleaks	On	On
report_zend_debug	On	On
request_order	GP	GP
safe_mode	Off	Off
safe_mode_exec_dir	*no value*	*no value*
safe_mode_gid	Off	Off

Scan down the output listing until you get to the Core settings for PHP and in this table you'll find the setting for REGISTER_GLOBALS. First column is the local value and the second column is the master value (originally set implementation value)

Using PHP With REGISTER_GLOBALS OFF

- When REGISTER_GLOBALS is set to OFF you must receive XHTML form input data using the $_POST, or $_GET associative arrays.
- Go back and look at the PHP scripts on pages 25 and 30 and you will see the $_POST associative array has been used in both examples to extract the input form data.
- We’ll deal with this in more detail later, but for now these two examples should give you a good idea of how form data extraction is handled in PHP scripts.
- There are many other associative arrays utilized in PHP. The remainder of this set of notes is devoted to some of these arrays.

Viewing Client/Server Environment Variables

- Knowledge of a client's execution environment is useful to system administrators who want to provide client-specific information.

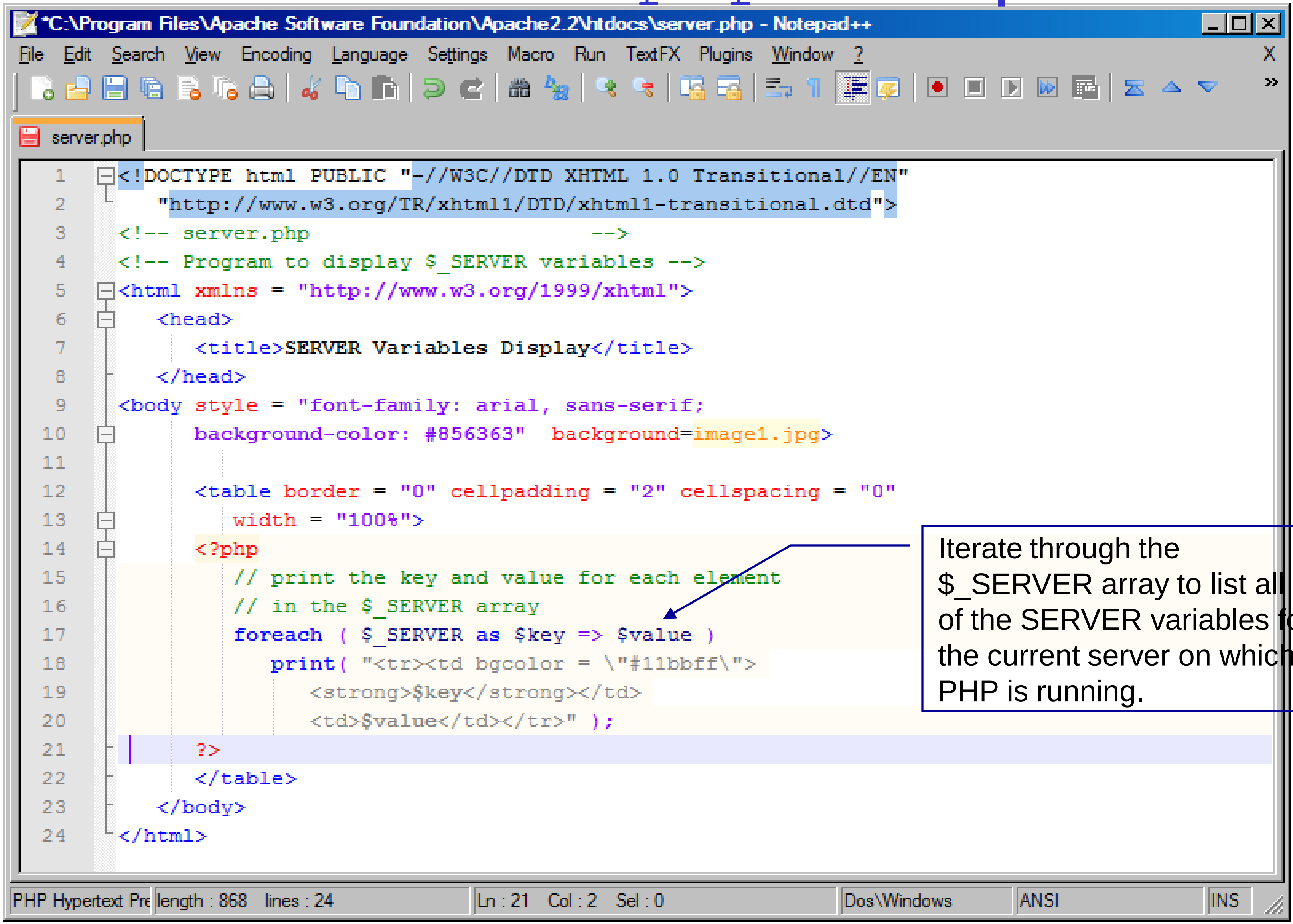
- Environment variables contain information about a script's environment, such as the client's web browser, the HTTP host and the HTTP connection.
  - The table on the next page summarizes some of the superglobal arrays defined by PHP.

- The XHTML document on page 37 displays the values of the server's environment variables in a table. PHP stores the server variables and their values in the `$_SERVER` array. Iterating through the array allows one to view all of the server's environment variables.

Some Superglobal Environment Arrays

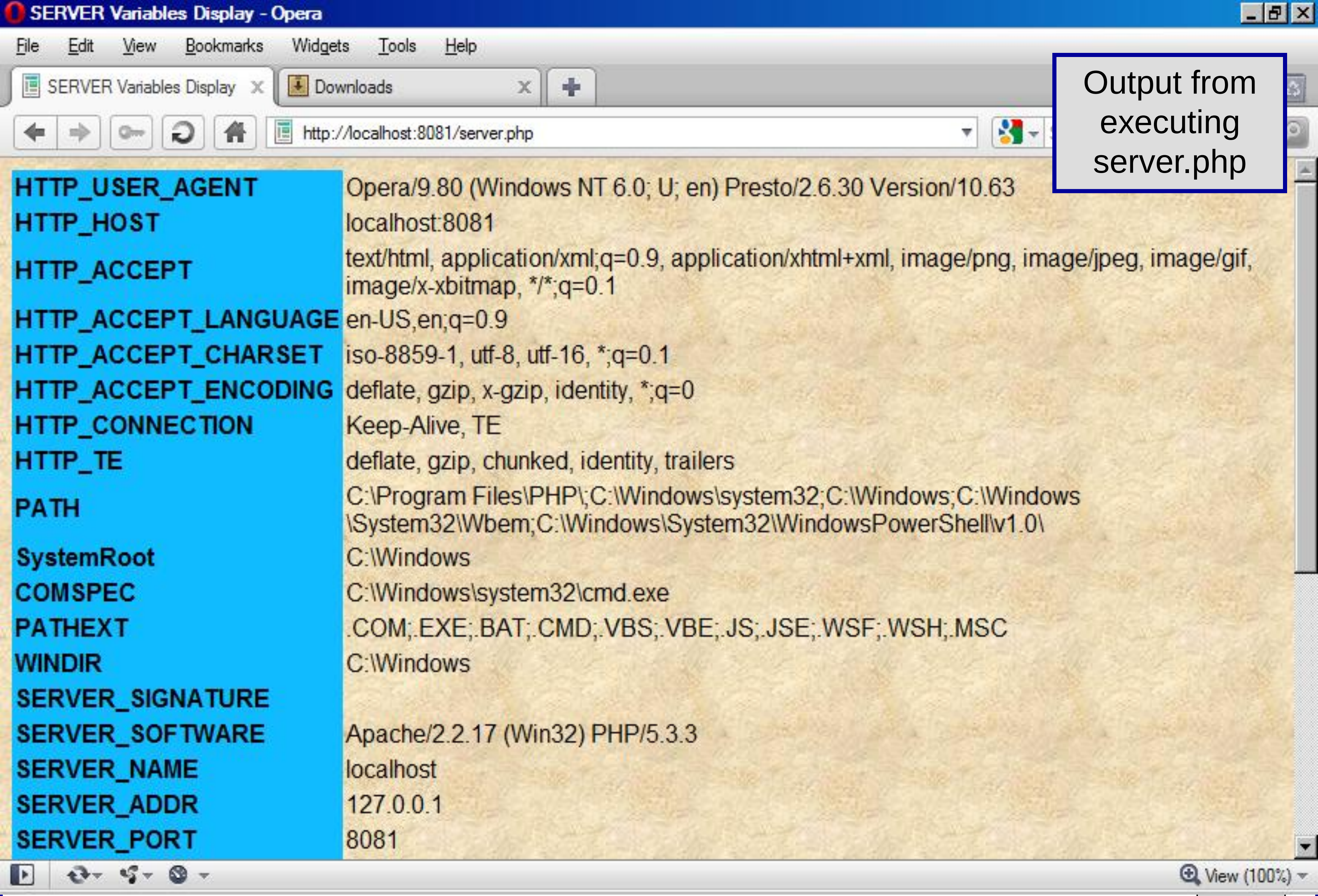
Variable Name	Description
$_SERVER	Data about the currently running server.
$_ENV	Data about the client's environment.
$_GET	Data posted to the server by the `get` method.
$_POST	Data posted to the server by the `post` method.
$_COOKIE	Data contained in cookies on the client's computer.
$GLOBALS	Array containing all global variables.

server.php Example


*C:\Program Files\Apache Software Foundation\Apache2.2\htdocs\server.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
server.php
PHP Hypertext Pre
length : 868 lines : 24
Ln : 21 Col : 2 Sel : 0
Dos\Windows
ANSI
INS

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!-- server.php                      -->
<!-- Program to display $_SERVER variables -->
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>SERVER Variables Display</title>
   </head>
<body style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1.jpg>

      <table border = "0" cellpadding = "2" cellspacing = "0"
         width = "100%">
      <?php
         // print the key and value for each element
         // in the $_SERVER array
         foreach ( $_SERVER as $key => $value )
            print( "<tr><td bgcolor = \"#11bbff\">
               <strong>$key</strong></td>
               <td>$value</td></tr>" );
      ?>
      </table>
   </body>
</html>
```

Iterate through the $_SERVER array to list all of the SERVER variables for the current server on which PHP is running.



HTTP_USER_AGENT	Opera/9.80 (Windows NT 6.0; U; en) Presto/2.6.30 Version/10.63
HTTP_HOST	localhost:8081
HTTP_ACCEPT	text/html, application/xml;q=0.9, application/xhtml+xml, image/png, image/jpeg, image/gif, image/x-xbitmap, */*;q=0.1
HTTP_ACCEPT_LANGUAGE	en-US,en;q=0.9
HTTP_ACCEPT_CHARSET	iso-8859-1, utf-8, utf-16, *;q=0.1
HTTP_ACCEPT_ENCODING	deflate, gzip, x-gzip, identity, *;q=0
HTTP_CONNECTION	Keep-Alive, TE
HTTP_TE	deflate, gzip, chunked, identity, trailers
PATH	C:\Program Files\PHP\;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\
SystemRoot	C:\Windows
COMSPEC	C:\Windows\system32\cmd.exe
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
WINDIR	C:\Windows
SERVER_SIGNATURE	
SERVER_SOFTWARE	Apache/2.2.17 (Win32) PHP/5.3.3
SERVER_NAME	localhost
SERVER_ADDR	127.0.0.1
SERVER_PORT	8081

Output from executing server.php