

EECS1012

Net-centric Introduction to Computing

Crash Course on PHP

Acknowledgements

Contents are adapted from web lectures for “Web Programming Step by Step”, by M. Stepp, J. Miller, and V. Kirst.

Slides have been ported to PPT by Dr. Xenia Mountroudou.

These slides have been edited for EECS1012, York University.

The contents of these slides may be modified and redistributed, please give appropriate credit.

(Creative Commons) Michael S. Brown, 2018.

PHP

2

- This material will **not** be on the exam, subject matter test, lab test, etc
- We show this, just to make you aware of client side programming and how similar languages are
- Now that you understand JavaScript, you will be surprised how much PHP you will understand

3

Server side basics

URLs and web servers

4

```
http://server/path/file.html
```

- When you type a URL in your browser:
 - Your computer looks up the server's IP address using DNS
 - Your browser connects to that IP address and requests the given file
 - The web server software (e.g. Apache) grabs that file from the server's local file system
 - The server sends back its contents to you

URLs and web servers (cont.)

5

```
http://www.facebook.com/home.php
```

- **Some URLs actually specify programs** that the web server should *run*, and then send the output of these program to you as the result:
 - ▣ The above URL tells the server **facebook.com** to run the program **home.php** and send back its output (which is an HTML document)

Server-Side web programming

6

- Server-side pages are programs written using one of many web programming languages/frameworks
 - ▣ examples: PHP, Java/JSP, Ruby on Rails, ASP.NET, Python, Perl



Server-Side web programming (cont.)

7

- Also called *server side scripting*:
 - **Dynamically** edit, change or add any content to a Web page before sending a browser (server-side)
 - Respond to user queries or data submitted from HTML forms (*this is coming soon – lecture on Forms*)
 - Access any data or databases and return the results to a browser
 - Customize a web page to make it more useful for individual users
 - Provide security since your **server code** cannot be viewed from a browser

What is PHP?

8

- PHP stands for "PHP Hypertext Preprocessor"
- Server-side scripting language
- **Used to make web pages dynamic:**
 - ▣ provide different content depending on context
 - ▣ interface with other services: database, e-mail, etc.
 - ▣ authenticate users
 - ▣ process form information
- PHP code can be embedded in HTML5 code



PHP history

9

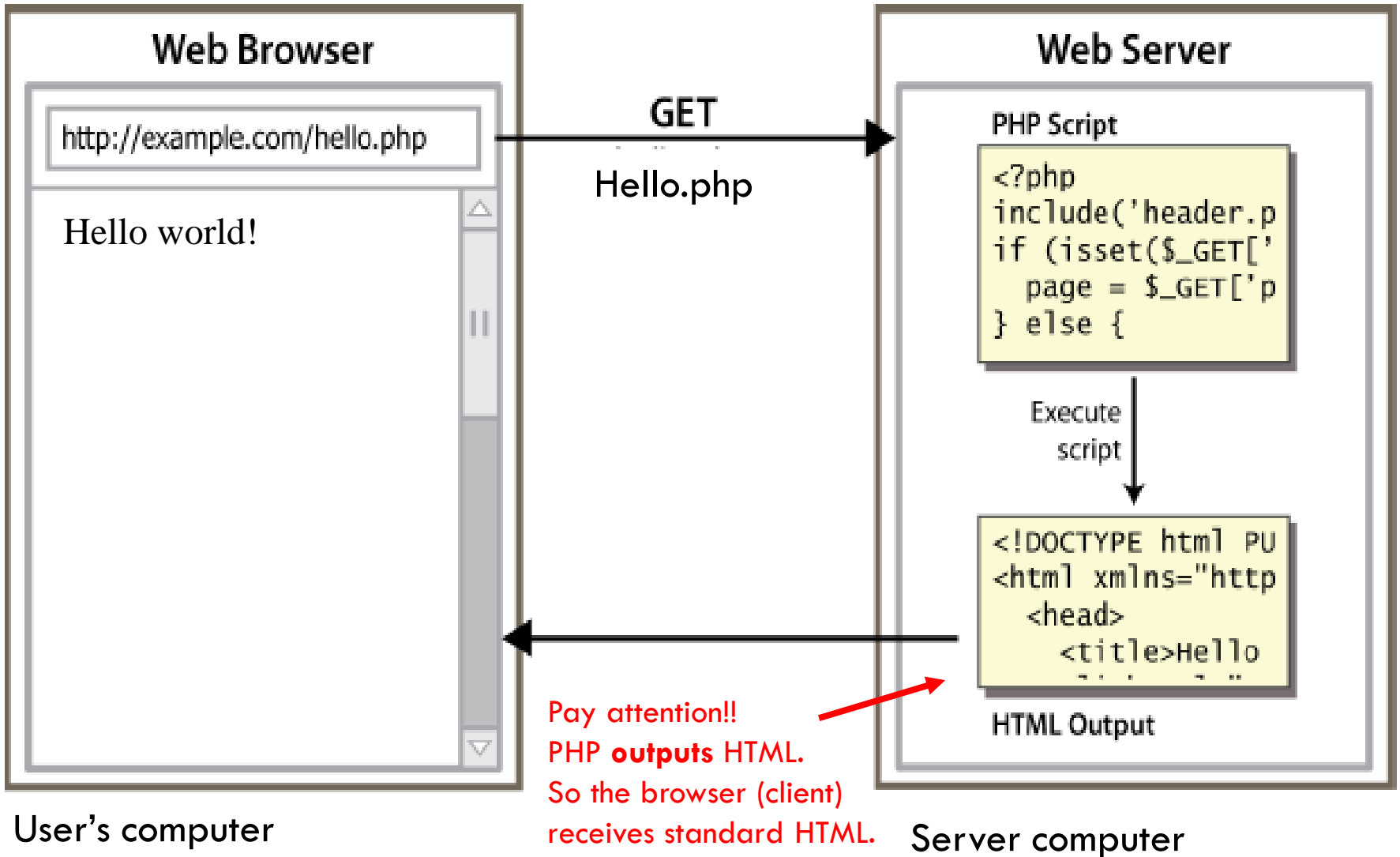
- Created in 1994 by Rasmus Lerdorf, originally called “Personal Home Page” language
 - ▣ He moved to Canada in 1980 from Denmark
- He did not intend PHP to become a new programming language
- He open sourced it and it grew on its own
- Rasmus graduated from University of Waterloo in Applied Science in System Engineering



Rasmus Lerdorf
(Canadian!)

Lifecycle of a PHP web request

10



'Hello, world!' PHP program

11

```
<html>
<head> </head>
<body> <p>
  <?php
    print "Hello, world!";
  ?>
</p> </body>
</html>
```

PHP

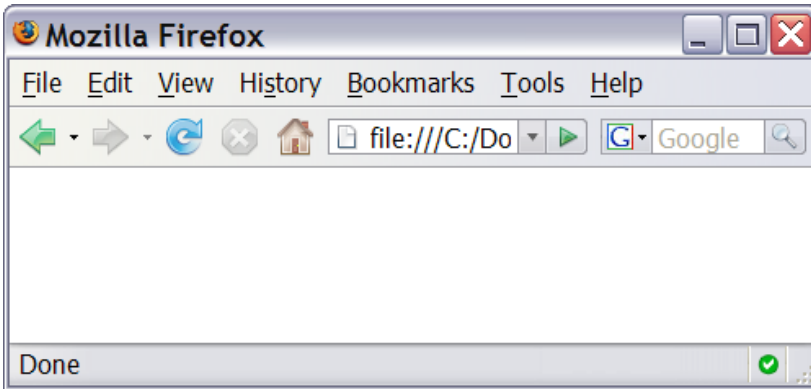
Hello, world!

output

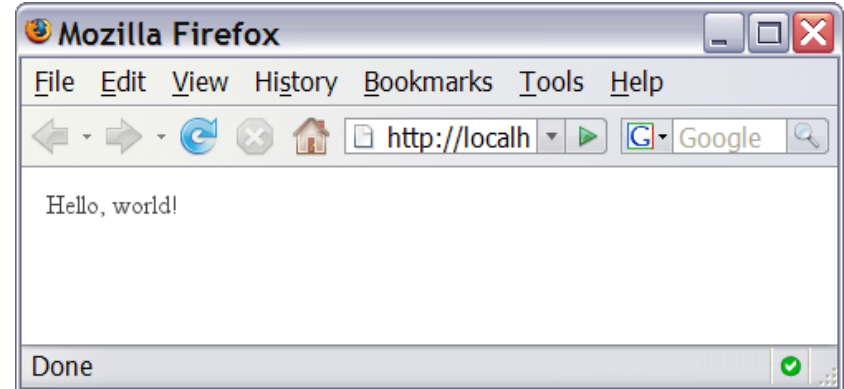
- ❑ **PHP is integrated into HTML!!!!**
- ❑ The print statement will output the words **Hello, world!** directly in the HTML file.
- ❑ The command “echo” can be used just like print, you may see echo used in other resources (e.g. by the w3school)

Viewing PHP output in Browser

12



Unlike JavaScript, you **cannot** view PHP directly through your browser if you were to load it up as a file. You must use a **web server** that supports PHP.



This example uses my virtual machine application and accesses "localhost". As a result I'm contacting the local web server.

13

PHP Basic Syntax

PHP syntax template

14

HTML content

<?php

PHP code

?>

HTML content

<?php

PHP code

?>

HTML content ...

PHP

- Contents of a .php file between `<?php` and `?>` are executed as PHP code . . output from PHP is injected into the HTML code
- All other contents are output as pure HTML
- We can switch back and forth between HTML and PHP "modes"

Simple PHP Example

15

```
<!DOCTYPE html>
<html>
<head></head>
<body>
  <h1> <?php print "Output from PHP"; ?> </h1>
  <p> This is a simple example of PHP </p>
  <p> The square root of 5 is <?php print sqrt(5); ?> </p>
</body>
</html>
```

PHP

```
<!DOCTYPE html>
<html>
<head></head>
<body>
  <h1> Output from PHP </h1>
  <p> This is a simple example of PHP </p>
  <p> The square root of 5 is 2.2360679774998 </p>
</body>
</html>
```

OUTPUT

PHP print

16

Print command is the main way to generate output in PHP.

```
print "Hello, World! \n";  
print "Escape \"chars\" need a \\ before them !\n";  
print "You can have  
line breaks in a string."  
print 'A string can use "single-quotes". It\'s cool!';
```

PHP

```
Hello, World!  
Escape "chars" need a \ before them !  
You can have  
line breaks in a string  
A string can use "single-quotes". It's cool!
```

PHP output

Print escape characters

17

- <http://phppot.com/php/php-escape-sequences/>

Escape Sequence	Description
<code>\t</code>	Insert a tab in the text at this point.
<code>\b</code>	Insert a backspace in the text at this point.
<code>\n</code>	Insert a newline in the text at this point.
<code>\'</code>	Insert a single quote character in the text at this point.
<code>\"</code>	Insert a double quote character in the text at this point.
<code>\\</code>	Insert a backslash character in the text at this point.

CS

SEE – Special Characters just like JavaScript!!!

Print with double " vs single quotes '

18

```
<?php
print "You need an escape sequence to print \"  \n";
print "But you don't to print single quotes '  \n";
print 'You need an escape sequence to print \'  \n';
print 'But you don\'t a print a double quote "  \n';
?>
```

PHP

You need an escape sequence to print "
But you don't to print single quotes '
You need an escape sequence to print '
But you don't a print a double quote "

PHP Output

Output: **Note, what is shown above is not what is seen in the browser. This is what PHP outputs. This will be injected into your HTML page.**

Example (PHP code)

19

```
<!DOCTYPE html>
<html>
<head>
    <title>My First PHP Page</title> </head>
<body>
<p>
    <?php
        print "Hello, World!\n";
        print "Escape \"characters\" are the same as in Java!\n";
        print "You can have
                line breaks in a string\n";
        print 'A string can use "single-quotes". It\'s cool!';
    ?>
</p>
</body>
</html>
```

PHP

Example (HTML output: source)

20

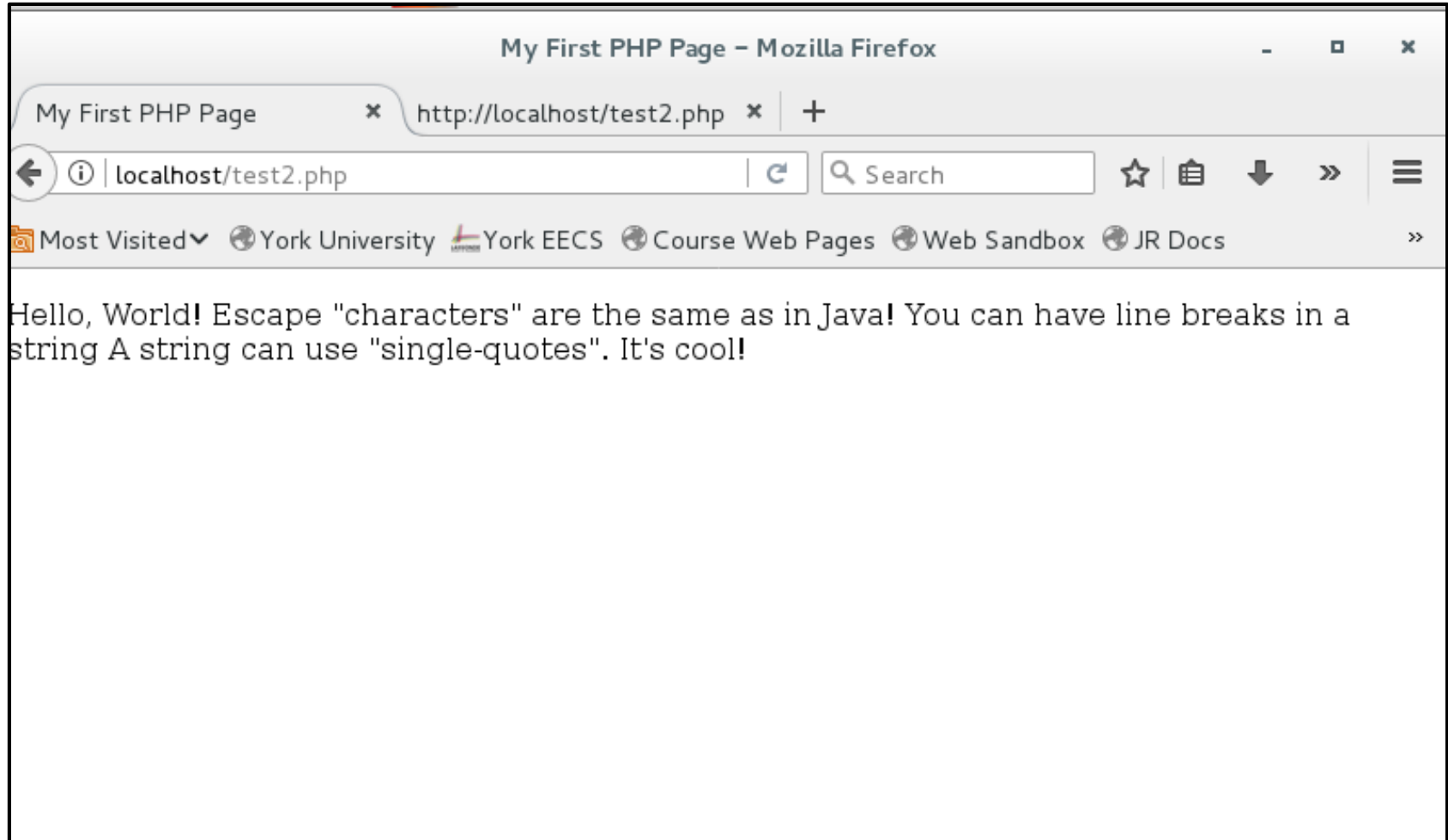
```
<!DOCTYPE html>
<html>
  <head>
    <title>My First PHP Page</title>
  </head>

  <body>
    <p>
      Hello, World!
      Escape "characters" are the same as in Java!
      You can have
      line breaks in a string
      A string can use "single-quotes". It's cool!
    </p>
  </body>
</html>
```

HTML OUTPUT

Example: (Browser rendering)

21



PHP ⇒ HTML ⇒ Browser

22

- Keep in mind that PHP code outputs text that is injected into the HTML page
- The HTML page is then interpreted (or rendered) by the browser
- These lecture notes are an “introduction to PHP”, so we will not be looking at the browser output (yet), mainly just the PHP output

Data Types

23

- Just like JavaScript, PHP has data types
- Like JavaScript, PHP is known as a “loosely typed” language. That means it decides the data type dynamically. This means variables storing data can change types.

Data Types

24

TYPE	Explanation	Example
int	A variable that stores whole number values (that is, an integer)	1, 3, -1, 0, 100
float	A variable that stores real numbers. For example, $1/3 = 0.3333$ is a float.	1.22, 10.99, -10.34934
boolean	A variable that wholes only two possible values – true or false .	true or false (1 or 0)
string	A variable that is a collection of characters, we call this a string	“Hello”, “EECS1012”, “Deaner”
array	A variable that is actually a collection of variables that can be access with an index (or a key)	[1,2,3,4, ...] [“hello”, “deaner”, ...] [3.4, 3333.4, -1.344, ...]
object	A bit more complex, out of the scope of this class.	More complex
NULL	Special type that has the value of “NULL” (that is computer speak for “nothing”. We sometimes equated with “false”).	Hard to give an example. We probably won't use this in our class.

Variables

25

```
$name = some expression;
```

```
$user_name = "mundruid78";  
$age = 16;  
$drinking_age = $age + 3;  
$this_class_rocks = TRUE;
```

PHP

- In programming, a variable is used to store information
- Data is assigned to a variable using an = symbol

Major difference with JavaScript is that PHP variables **must** start with a \$. This is common in several scripting languages (PERL, Bash, so on).

PHP variable names

26

PHP variable name rules

- A variable starts with the \$ sign, followed by the name of the variable
- A variable name **must** start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (\$age, \$Age, and \$AGE are considered three different variables)

Quite similar to JavaScript, eh?

PHP and variable types

27

- PHP automatically assigns the type for a variable
- PHP will also change the type dynamically

```
$a = 1;           # $a is a integer type  
$b = "12";       # $b is a string type (char '1' and '2')  
$a = "Poutine";  # now $a is a string type
```

PHP

Printing variables

28

```
$a = 10;      # integer type
print "The value of our variable is $a. \n";
print $a;    # you don't have to put this in quotes
```

PHP

```
The value of our variable is 10.
10
```

output

You can use `print` (or `echo`) to output the value of the variable. This will be used very frequently in PHP.

Expressions (and statements)

29

- An expression is the combination of one or more variables, values, operators, or functions that computes a result.

```
$num1 = 5;           # value 5 is the expression

$num2 = $num1 + 10;  # $num1 + 10 is the expression,
                    # operator is +, this computes 5 + 10;

$num2 = $num2 + 1;   # this uses $num2 and assigns the
                    # result back to $num

$str1 = "hello";     # value is "hello"
$str2 = "world";     # value is "world"

$num1 = ((3.14) * 10.0) / 180.0; # multiple operators
```

PHP syntax breakdown

30

A variable

```
$num1 = ((3.14) * 10.0) / 180.0;
```

= is an assignment.
This takes the result of the expression and makes it the current value of the variable on the left side of the assignment.

An expression that will be evaluated to compute some result. Later we will see this can be other things, like calling a “function”, or testing if something is “true” or “false”

semicolon.
in PHP, we will use a semicolon to end most of our statements.

This is identical to our slide on JavaScript! Only difference, that variable has a \$.

Basic arithmetic operators

31

$\$a + \b	Addition	Sum of $\$a$ and $\$b$.
$\$a - \b	Subtraction	Difference of $\$a$ and $\$b$.
$\$a * \b	Multiplication	Product of $\$a$ and $\$b$.
$\$a / \b	Division	Quotient of $\$a$ and $\$b$.
$\$a \% \b	Modulo	Remainder of $\$a$ divided by $\$b$.

Here $\$a$ and $\$b$ could be variables, but we could also replace them with numbers. $10 + 20$, $3.14 / 2.0$, etc. ..

Evaluation and assignment

32

```
1: $num1 = 10;  
2: $num1 =  $\underbrace{\$num1 + 10}$ ;
```

PHP will interpret these statements as:

1: \$num1 is assigned 10

2: (\$num1) + 10

10 + 10

20

\$num1 is assigned 20

The expression is always computed **before** the assignment. This allows us to use a variable and assign the result back to the same variable.

“Short hand” assignment operators

33

Assignment

`$a += $b;`

`$a -= $b;`

`$a *= $b;`

`$a /= $b;`

`$a %= $b;`

`$a++;`

`$a--;`

Same as:

`$a = $a + $b;` Addition

`$a = $a - $b;` Subtraction

`$a = $a * $b;` Multiplication

`$a = $a / $b;` Division

`$a = $a % $b;` Modulus

`$a = $a + 1;` Self Addition

`$a = $a - 1;` Self subtraction

PHP Operator Precedence (Math)

34

```
$num1 = 5 * 5 + 4 + 1 / 2;    # What is the answer?  
$num2 = 5 * (5 + 4) + 1 / 2; # What is the answer?  
print "$num1 and $num2";
```


PHP

29.5 and 45.5

output

Operator Precedence

() Highest
* / %
+ - Lowest



Compute results based on order of precedence, and from left to right in the expression.

Example from previous slide

35

$$\begin{aligned} \$num1 &= 5 * 5 + 4 + 1 / 2; \\ &\quad \underbrace{\hspace{1.5cm}} \hspace{1.5cm} \underbrace{\hspace{1.5cm}} \\ &\quad (a) \ 25 \hspace{10cm} 0.5 \ (b) \\ &\quad \quad \underbrace{\hspace{3cm}} \\ &\quad \quad (c) \ 29 \hspace{5cm} \underbrace{\hspace{3cm}} \\ &\quad \quad \quad \underbrace{\hspace{6cm}} \\ &\quad \quad \quad (d) \ 29.5 \end{aligned}$$

Based on operator precedence, the expression would be computed in the following order:

- (a) $5 * 5 = 25$
- (b) $1 / 2 = 0.5$
- (c) $(a) + 4$ [where (a) is 25]
- (d) $29 + (b)$ [where (b) is 0.5]

final 29.5

$$\begin{aligned} \$num1 &= 5 * (5 + 4) + 1 / 2; \\ &\quad \quad \underbrace{\hspace{1.5cm}} \\ &\quad \quad (a) \ 9 \\ &\quad \quad \quad \underbrace{\hspace{1.5cm}} \\ &\quad \quad \quad (b) \ 45 \hspace{10cm} \underbrace{\hspace{1.5cm}} \\ &\quad \quad \quad \quad \underbrace{\hspace{3cm}} \\ &\quad \quad \quad \quad (d) \ 45.5 \end{aligned}$$

Based on operator precedence, we would have:

- (a) $(5+4) = 9$
- (b) $5 * (a)$ [where (a) is 9]
- (c) $1 / 2 = 0.5$
- (d) $(b) + (c)$ [$45 + 0.5$]

Final 45.5

EXACTLY THE SAME AS JAVASCRIPT!!!!

PHP functions and function calls

36

- One of the most powerful aspect of PHP is the large variety of built-in functions that can be used (over 1000 functions)
- A function is a procedure or routine that performs a task and (generally) returns a value. Functions can also take parameters as part of its “call”.

```
$num = rand();  
    # function rand() returns a random number  
$num = rand(1,10);  
    # function rand() returns a random number between 1-10
```

PHP

PHP function call breakdown

37

Most functions return a value that can be assigned to a variable or used in an expression.



```
$a = rand();
```

The code snippet is enclosed in a light blue box with a dark blue border. A blue arrow points from the text above to the opening parenthesis of the function call.

function name.

When PHP sees **()**, it searches for a function with that name. Note that the function does not have a \$ in front of its name.



```
$a = rand(1, 100);
```

The code snippet is enclosed in a light blue box with a dark blue border. A blue arrow points from the text below to the opening parenthesis of the function call.

function parameters

Most functions allow you to "pass" parameters to the function that will be used when computing the result. The parameters are placed within the parenthesis.

Useful PHP math functions

38

Function	Description
<code>abs(n)</code>	absolute value
<code>ceil(n)</code> , <code>floor(n)</code>	ceil means round up, floor means round down
<code>log(n)</code>	compute the natural logarithm
<code>min</code> , <code>max(a, b, ..)</code>	min or max of a sequence of numbers: e.g. <code>max(50, 43, 1, -1, 30) = 50</code>
<code>pow(base, exp)</code>	compute (exponent) base^{exp}
<code>rand()</code> , <code>rand(min,max)</code>	return a random number, or a random number within a min-max range
<code>round(n)</code> <code>round(n, digits)</code>	round a number to the nearest integer or decimal place
<code>sqrt(n)</code>	square root

JavaScript relied on an object to provide this functionality. `Math.abs()`, `Math.ceil()`. PHP is not as “object-oriented” as JavaScript.

Math function examples

39

math funcs	result
<code>\$a = -50;</code> <code>\$b = 25;</code> <code>\$c = 32;</code> <code>\$d = 3.493;</code>	Result
<code>abs(\$a)</code>	50
<code>ceil(\$d)</code>	4
<code>floor(\$d)</code>	3
<code>log(\$c)</code>	3.4657359027997
<code>min(\$b, \$c, \$d)</code>	3.493
<code>max(\$b, \$c)</code>	32
<code>rand()</code>	1659778700*
<code>rand(1, 10)</code>	8*
<code>round(\$d)</code>	3
<code>round(\$d, 1)</code>	3.5
<code>sqrt(\$b)</code>	5

*result will be a large random number

*result will be a random number between 1-10

Variables auto conversion

40

- PHP converts between types automatically in many cases:
 - ▣ string → int auto-conversion on +
 - ▣ int → float auto-conversion on / (division)

```
<?php
$num = "5";           # this is a string type
$num2 = $num + 1;    # the string was converted to an integer!
print "Result #1 is $num2 \n";
$num2 = $num2 / 5;   # integer was converted to a float
print "result #2 is $num2 \n";
?>
```

PHP

```
Result #1 is 6
Result #2 is 1.2
```

output

Sometime called “type juggling”

41

```
<?php
$foo = "1"; // $foo is string
$foo *= 2; // $foo is now an integer (2)
$foo = $foo * 1.3; // $foo is now a float (2.6)
$foo = 4 / 2; // $foo is still an integer (2)
$foo = 3 / 2; // $foo is now a float (1.333333)
```

```
?>
```

PHP

- **PHP will try to convert a string to a numerical value if it finds numerical content in the string**
- **This is the opposite to JavaScript. JS always converted numbers to strings. These are the subtle difference you have to learn between languages.**

Comments

42

```
# single-line comment
// single-line comment
/*
multi-line comment
*/
```

PHP

- Allows several different types of comments
- `/* */` is PHP and JS style comments
- `//` is PHP and JS style
- `#` is used in PHP
 - a lot of PHP code uses `#` comments

43

Strings

String Type

44

```
$favorite_food = "Ethiopian";  
$favorite_food = $favorite_food . " cuisine";  
print $favorite_food;
```

PHP

- String are used extensively in web programming, because it is the main variable for storing “text”
- Strings can be “added” together, we call this *concatenation*
- string concatenation operator is . (period), not +
 - ▣ "Hel" . "lo" -> "Hello"
 - ▣ 5 . "2 turtle doves" -> "52 turtle doves"

Simple String example

45

```
<?php
$name = "Abdel Zhang"; # string using double quotes
$degree = 'EECS'; # string using single quotes
$course = $degree . "1012"; #concatenation operator

print " $name is taking $course \n "; # printing

?>
```

PHP

Abdel Zhang is taking EECS1012


output

String concatenation operator

46

Operator Precedence

() Highest
* / %
+ - . Lowest



The concatenation operator . has the same precedence as + and -. Given type juggling, it can sometimes be tricky to see what is happening.

Expression	Result
1 + "3"	4
1 + "3 French hens"	4
1 . "2"	"12"
1 + 3 + "5" + 7 + 9	25
1 + "not a number"	1
1 . 3 . "5" . 7 . 9	"13579"
(1 + 3) . "5" + (7 + 9)	"4516"

JavaScript used the + operator to mean string concat. PHP uses the "." operator (not to be confused with accessing an object in JS)

String length?

47

Expression	Result
<code>strlen(<i>s</i>)</code>	Returns length (number of characters) of string <i>s</i>

Unlike JavaScript, PHP variables are not objects. So there is no associated property called “length”. Instead, we need to call a function that returns the length of a string. See next slides.

String indexing []

48

```
$str1 = "J. Trudeau";
```

Index	0	1	2	3	4	5	6	7	8	9
Character	J	.		T	r	u	d	e	a	u

Expression	Result
<code>\$str1[0]</code>	"J"
<code>\$str1[3]</code>	"T"
<code>\$str1[2]</code>	" " (space character)
<code>strlen(\$str1)</code>	10 (be careful - why 10?)
<code>\$str1[strlen(\$str1) - 1]</code>	"u"

WOW – EXACTLY LIKE JAVASCRIPT!!!!

Interpreted Strings

49

- Strings with variables written directly inside double quotes will be inserted as the program runs

```
$first = "Abdel";  
$last = "Zhang";  
$number = 1223043;  
$student = "$first , $last $number ";  
print $student ;
```

PHP

Abdel, Zhang 1223045

output

50

Control Statements

PHP – comparison operators

51

Operator	Name	Example	Result	W3School
==	Equal	<code>\$x == \$y</code>	Returns true if \$x is equal to \$y	Show it »
===	Identical	<code>\$x === \$y</code>	Returns true if \$x is equal to \$y, and they are of the same type	Show it »
!=	Not equal	<code>\$x != \$y</code>	Returns true if \$x is not equal to \$y	Show it »
<>	Not equal	<code>\$x <> \$y</code>	Returns true if \$x is not equal to \$y	Show it »
!==	Not identical	<code>\$x !== \$y</code>	Returns true if \$x is not equal to \$y, or they are not of the same type	Show it »
>	Greater than	<code>\$x > \$y</code>	Returns true if \$x is greater than \$y	Show it »
<	Less than	<code>\$x < \$y</code>	Returns true if \$x is less than \$y	Show it »
>=	Greater than or equal to	<code>\$x >= \$y</code>	Returns true if \$x is greater than or equal to \$y	Show it »
<=	Less than or equal to	<code>\$x <= \$y</code>	Returns true if \$x is less than or equal to \$y	Show it »

WOW .. Exactly like JavaScript!!!!

If example

52

Example

```
if ($grade == "A")
{
    print "I LOVE EECS1012 \n";
    print "It is my favorite class \n";
}
```

PHP

Exactly like JavaScript!

If/else example

53

Example

```
if ($grade == "A")
{
    print "I LOVE EECS1012 \n";
    print "It is my favorite class \n";
}
else
{
    print "I HATE EECS1012 \n";
    print "It is my least favorite class \n";
}
```

PHP

Exactly like JavaScript!

While Loop

54

```
$i=0;
while ($i < 10)
{
    print "$i squared is " . $i * $i . ".\n";
    $i++; # add 1 to $i, put result back in $i
}
```

PHP

0 squared is 0.
1 squared is 1.
2 squared is 4.
3 squared is 9.
4 squared is 16.
5 squared is 25.
6 squared is 36.
7 squared is 49.
8 squared is 64.
9 squared is 81.

Exactly like JavaScript!

for/loops

55

```
for (initialization; condition; update) {  
    statements;  
}
```

PHP

```
for ($i = 0; $i < 10; $i++) {  
    print "$i squared is " . $i * $i . ".\n";  
}
```

PHP

Exactly like JavaScript!

56

Arrays

Arrays

57

```
<?php
    $food = array("falafel", "pide", "poutine");
    print "I like to eat $food[0] and $food[2] ";
?>
```

PHP

I like to eat falafel and poutine.

output

- Notation is slightly different than JS.
- We need to use an array() function.

**Functionality, exactly like JS.
Some differences.**

Indexed Arrays

58

```
<?php
    $food = array("falafel", "pide", "poutine");
    print "My favorite is $food[0] \n";
?>
```

PHP

- **Index arrays** (or numerical arrays) are arrays where the individual values in the array are accessed with a *numeric index*. Indexing starts at position 0, not 1.

\$food [index]	0	1	2
array value	"falafel"	"pide"	"poutine"

Indexed array syntax

59

\$var[**index**]

Array variable name

A diagram illustrating the indexed array syntax. At the top center, the text "\$var[index]" is displayed. The "\$var" part is in blue, and the "index" part is in red. Below this, two blue arrows point upwards. The left arrow points to the "\$var" part, and the right arrow points to the "index" part. Below the left arrow is the text "Array variable name". Below the right arrow is the text "index (sometimes called 'offset') that you want to access within brackets []".

index (sometimes called "offset")
that you want to access
within brackets []

Array() function

60

```
<?php
$num = array(100, 90, 80, 70, 60, 50, 40, 30, 20, 10);
print $num[0]; # output would be 100
?>
```

PHP

- The function `array()` can be used to create an array variable as shown above. In this example, the data in the array are integers.

\$num [index]	0	1	2	3	4	5	6	7	8	9
array value	100	90	80	70	60	50	40	30	20	10

Indexed arrays are similar to how we accessed individual characters in string variables.

count function for arrays

61

```
$a = array("Pide", "Dosa", "Falafel", "Poutine");  
$a_length = count($a);  
print "The number of items in the array is $a_length. \n";
```

PHP

The number of items in the array is 4.

output

- `count(a)` returns the number of elements in the array **a**. Sometimes we call this the "size" of the array, or "length" of the array.

JavaScript, we could access the `array.length` property, but in PHP, we need a function `count()`.

Manual array assignment

62

```
<?php
    $food[0] = "dosa";
    $food[1] = "pide";
    $food[2] = "poutine";

    print "I like to eat $food[0] and $food[2] ";
?>
```

PHP

I like to eat dosa and poutine.

output

- We can manually assign values to an array.
- This example did not use the array function, but the result is identical.

print_r function

63

```
<?php
    $array_var = array( "CSS", "PHP", "HTML", "Coding" );

    print_r($array_var);

?>
```

PHP

```
Array
(
    [0] => CSS
    [1] => PHP
    [2] => HTML
    [3] => Coding
)
```

output

This function is referred to as the "Print Readable" function and helps you visualize the contents of your data. You can use it with arrays or other data types.

Associative Arrays

64

```
<?php
    $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
    print $age["Peter"];
    print "\n";
    print $age["Joe"];
?>
```

PHP

```
35
43
```

output

- Associative arrays uses a “key” to access an individual element in the array.
- Syntax: `$var_name[key]`. The key is often a string, but can also be a numerical value.

Associative array syntax

65

\$var[**key**]

Array variable name

name of the key that you want
to access within brackets []

JavaScript does **not** support associate arrays.

Associative arrays using array()

66

```
$a = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
```

↑
key is
a string

↑
value is a string

```
$a = array(1 => "Five", 2 => "Two", 3 => "Three");
```

↑
key is
a number

↑
value is a string

Syntax to specify in the array() function: key => value. You will generally string values used as keys, but numerical values can also be used as shown above.

Associative array example

67

```
<?php  
  
$a["lecture"] = "Hall A";  
$a["labs"] = "William Small 106";  
$a["university"] = "York";  
$a["college"] = "Lassonde";  
print_r($a);  
?>
```

Array

```
(  
  [lecture] => Hall A  
  [labs] => William Small 106  
  [university] => York  
  [college] => Lassonde  
)
```

output

- You can also manually assign data to associative arrays.

Examples – array_keys()

68

```
$items= array("iPhone" => 988, "Samsung" => 700, "LG" => 500);  
  
$keys = array_keys( $items ); # returns an array with the keys from  
                                # items  
  
for($i=0; $i < count($keys); $i++)  
{  
    $key = $keys[$i];  
    print "Item: $i Brand: $key Price: $ $items[$key] \n";  
}
```

PHP

```
Item: 0 Brand: iPhone Price: $ 988  
Item: 1 Brand: Samsung Price: $ 700  
Item: 2 Brand: LG Price: $ 500
```

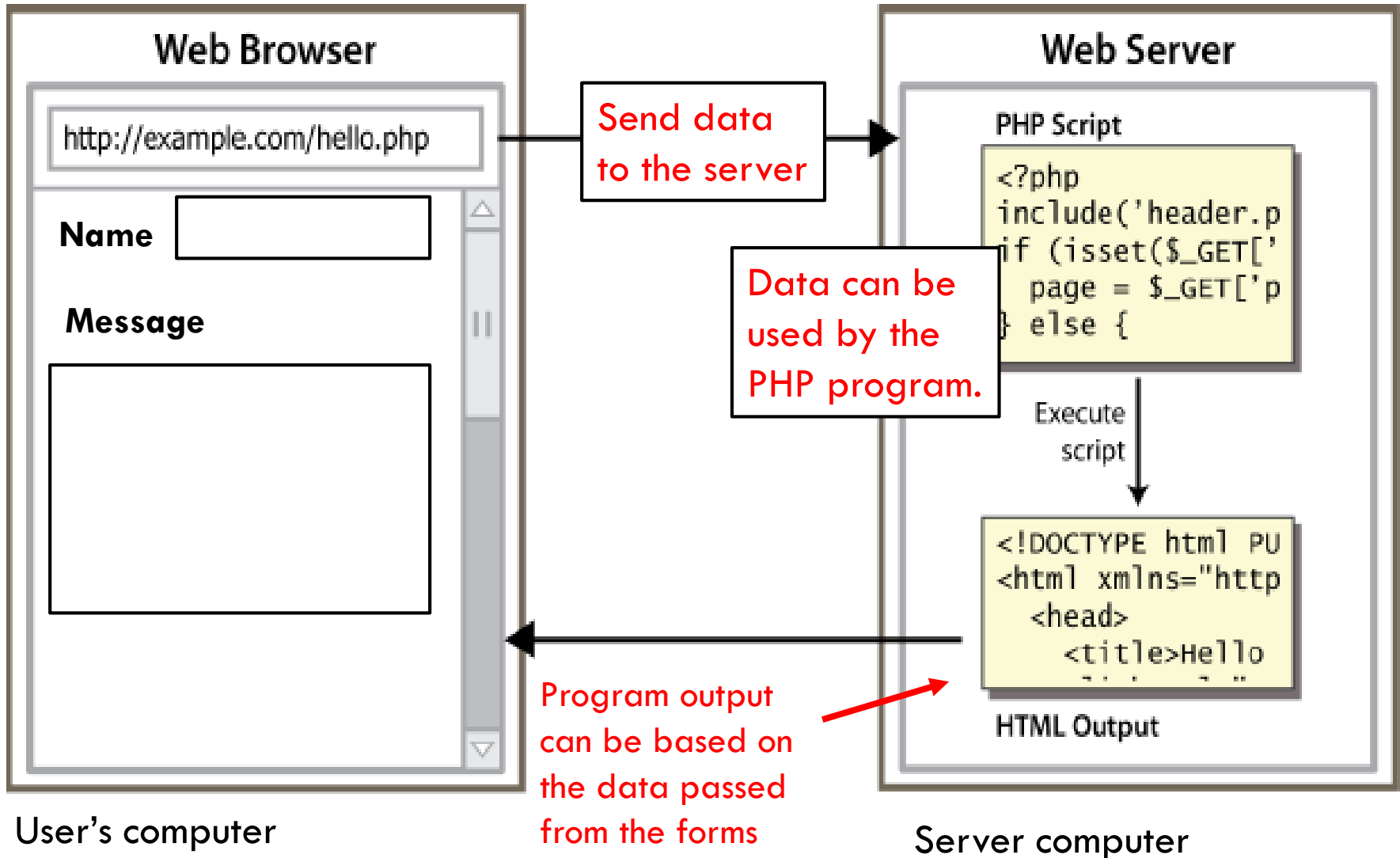
OUTPUT

69

Forms + PHP

Sending Data to a PHP program

70



Quick example

71

- Recall that a URL can have text after the file name:

- query string: a set of parameters passed to a web program

<http://www.google.com/search?q=poutine>

- parameter q is set to "poutine"

(from our first lecture)

PHP - \$_GET variable and URLs

72

`http://servername/filename.php?var1=value1 & var2=value2`



```
$_GET = array(var1 => value1, var2=> value2)
```

PHP has a special variable called "\$_GET" that stores the "query" values from the URL as an associative array. This can be access in the PHP program.

Example - \$_GET variable

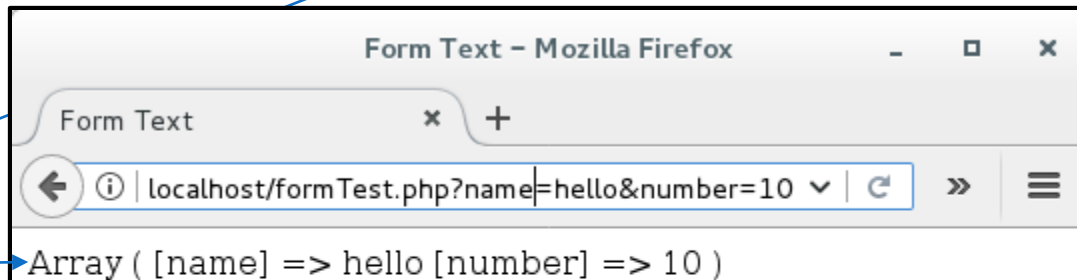
73

<http://localhost/formTest.php?name=hello&number=10>

formTest.php

```
<!DOCTYPE html>
<html>
<head>
  <title> Form Test</title>
</head>
<body>
  <?php
    print_r($_GET);
  ?>
</body>
</html>
```

`$_GET = array ("name"=>"hello", "number"=>"10")`



Form Text - Mozilla Firefox

Form Text x +

localhost/formTest.php?name=hello&number=10

Array ([name] => hello [number] => 10)

This simple PHP program prints out the contents of the `$_GET` variable (using `print_r`). See how the key and values of the associative array match the URL!! This provides a mechanism to pass information to PHP!

testForm.php

74

```
<!doctype html>
<html lang="en">
<head>
  <style>
    body { font-family: sans-serif;}
    tr, th, td { font-size: 1.25em; }
    table, td, tr, th { border: 1px solid black; }
    td, th { padding: 10px; }
  </style>
  <title> Test Form for EECS1012 </title>
  <meta charset="utf-8">
</head>
<body>
  <h1> Data sent from HTML Form </h1>

  <!-- continue on . Next slide →
```

testForm.php (con't)

75

```
<?php
  if (count($_GET) > 0)
  {
    print "<table> \n";
    print "<tr> <th> Name </th> <th> Value </th> </tr> \n";
    $keys = array_keys( $_GET ); # get keys from $_GET
    for($i=0; $i < count($keys); $i++)
    {
      $name = $keys[$i];
      $value = $_GET[$name];
      print "<tr><td>".htmlspecialchars($name).
        "</td><td>".htmlspecialchars($value)."</td></tr> \n";
    }
    print "</table> \n";
  }
  else {
    print "<h2> There was no data sent. </h2> \n";
  }
?>
</body>
</html>
```

PHP Summary

76

- Many similarities between JS and PHP
- PHP is not event driven
- It is also made to output directly to the HTML file, so lots of “print” statements, where JS didn’t have this.
- There are some subtle differences, e.g. no length properties for string and arrays
- PHP has associative arrays, JS does not
- Even though you do not know PHP, you can read some of the code.