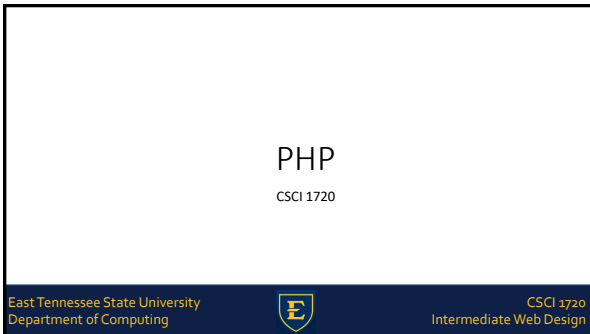
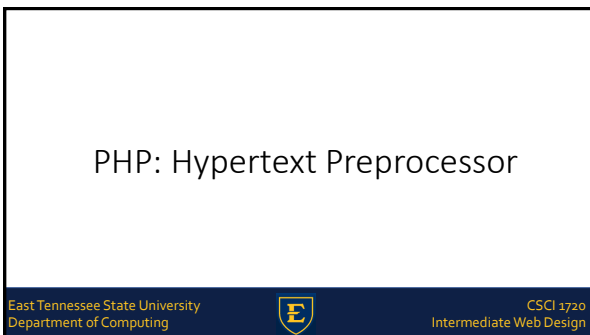


1



2



3


PHP: Hypertext Preprocessor

PHP is a server scripting language, and a powerful tool for making dynamic and interactive Web pages

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19043.1110]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>php --version
PHP 8.0.8 (cli) (built: Jun 29 2021 16:02:52) ( ZTS Visual C++ 2019 x64 )
Copyright (c) The PHP Group
Zend Engine v4.0.8, Copyright (c) Zend Technologies
C:\Windows\system32>
```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

4

PHP: Hypertext Preprocessor


PHP is an acronym for "PHP: Hypertext Preprocessor"

PHP is a widely-used, open source scripting language

PHP scripts are executed on the server

PHP is free to download and use

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

5

What is a PHP File?

PHP files can contain text, HTML, CSS, JavaScript, and PHP code

PHP code is executed on the server, and the result is returned to the browser as plain HTML

PHP files have extension ".php"

East Tennessee State University
Department of Computing




CSCI 1720
Intermediate Web Design

6

What Can PHP Do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design


7

What Can PHP Do?

- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data

With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML

East Tennessee State University
Department of Computing




CSCI 1720
Intermediate Web Design

8

Why PHP?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

East Tennessee State University
Department of Computing




CSCI 1720
Intermediate Web Design

9

PHP Syntax

A PHP script is executed by the server
 The result is then sent back to the browser
 A PHP script can be placed anywhere on a web page
 Scripts start with `<?php` and ends with `?>`

East Tennessee State University
 Department of Computing



CSCI 1720
 Intermediate Web Design


10

PHP Hypertext Preprocessor

```
<?php
// PHP code goes here
?>
```

The conventional file extension for a php file, oddly enough, is '.php'
 A PHP file normally contains HTML tags and some PHP scripting code

East Tennessee State University
 Department of Computing




CSCI 1720
 Intermediate Web Design

11

PHP Hypertext Preprocessor

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <h1>My first PHP page</h1>
  <?php
  echo "Hello World!";
  ?>
</body>
</html>
```

East Tennessee State University
 Department of Computing



CSCI 1720
 Intermediate Web Design


12

PHP Case Sensitivity

PHP isn't case-sensitive
By convention, use lower- and camelCase
Variable names ARE case sensitive:

```
$firstName != $Firstname
```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design


13

PHP Commenting

PHP supports several kinds of commenting

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

East Tennessee State University
Department of Computing




CSCI 1720
Intermediate Web Design

14

PHP Variables

A variable starts with a dollar sign '\$'
PHP is a non-declarative language – you don't tell it what kind of variable you're declaring
It'll figure it out when a value is assigned (or when it is initialized):

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

15

PHP Variables

```
<?php
$someNumber = 42;
?>
```

Number

```
<?php
$firstName = "Fred";
?>
```

String

East Tennessee State University
Department of Computing
CSCI 1720
Intermediate Web Design

16

Rules for PHP variables

- 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 and \$AGE are two different variables)

East Tennessee State University
Department of Computing
CSCI 1720
Intermediate Web Design

17

Output Variables

The PHP echo statement is often used to output data to the page

The following example will show how to output text and a variable

```
<p>
    &quot;Halt! 'oo goes there?&quot;;
</p>
<?php
    $whoGoesThere = " Arthur, King of the Britons";
    echo "I am $whoGoesThere!";
?>
```

→

```
"Halt! 'oo goes there?"
I am Arthur, King of the Britons!
```

East Tennessee State University
Department of Computing
CSCI 1720
Intermediate Web Design

18

Output Variables

You can also concatenate output


```

<p>
    &quot;Halt! 'oo goes there?&quot;;
</p>
<?php
    $whoGoesThere = " Arthur, King of the Britons";
    echo "I am " . $whoGoesThere . "!";
?>

```

"Halt! 'oo goes there?"
I am Arthur, King of the Britons!

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

19


PHP is a Loosely Typed Language

PHP automatically associates a data type to the variable, depending on its value

Since the data types are not set in a strict sense, you can do things like adding a string to an integer without causing an error

In PHP 7, type declarations were added. This gives an option to specify the data type expected when declaring a function, and by enabling the strict requirement, it will throw a "Fatal Error" on a type mismatch

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

20

PHP Variables Scope


In PHP, variables can be declared anywhere in the script.

The scope of a variable is the part of the script where the variable can be referenced/used

PHP has three different variable scopes:

- ▶ local
- ▶ global
- ▶ static

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

21

PHP global Keyword


The **global** keyword is used to access a global variable from within a function

```
<?php
$x = 5;
$y = 10;

function test() {
    global $x, $y;
    $y = $x + $y;
}

test();
echo $y; // outputs 15
?>
```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

22


PHP static Keyword

Normally, when a function is completed/executed, all of its variables are deleted

Sometimes we want a local variable NOT to be deleted. We need it for a further job

To do this, use the **static** keyword when you first declare the variable

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

23

PHP static Keyword


Note: The variable is still local to the function

```
<?php
function myTest() {
    static $x = 0;
    echo $x;
    $x++;
}

echo "I start out at ";
myTest();
echo "<br>Then I'm ";
myTest();
echo "<br>Now, I'm ";
myTest();
?>
```

I start out at 0
Then I'm 1
Now, I'm 2

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

24

PHP `echo` and `print` Statements

With PHP, there are two basic ways to get output: `echo` and `print`


We use these a lot in debugging code

More or less the same – `print` returns a value; `echo` doesn't

`print` can be used in expressions

`echo` is a little faster – can be used with or without parentheses

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

25

PHP `echo` and `print` Statements


```
<?php
$txt1 = "All that is gold does not glitter";
$txt2 = "Not all who wander are lost";

echo "<b>" . $txt1 . "</b>";
echo $txt2 . "<br>";
?>
```

All that is gold does not glitter

Not all who wander are lost

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

26

PHP `echo` and `print` Statements

The print statement can also be used with or without parenthesis

```
<?php
$txt1 = "All that is gold does not glitter";
$txt2 = "Not all who wander are lost";


print "<b>" . $txt1 . "</b>";
print $txt2 . "<br>";
?>
```

All that is gold does not glitter

Not all who wander are lost

Notice that we can include HTML markup in PHP output

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

27

PHP Data Types

PHP supports the following data types:

- ▶ String
- ▶ Integer
- ▶ Float (floating point numbers - also called double)
- ▶ Boolean
- ▶ Array
- ▶ Object
- ▶ NULL
- ▶ Resource



28

PHP Strings

As you no doubt know, a string is a series of characters

e.g. "Hello World!"

PHP has several functions associated with strings



29

PHP String Functions

strlen()

The PHP strlen() function returns the length of a string, e.g.,

```
<?php
strlen("Hello World!"); // returns 12
?>
```

or

```
<?php
$str = "Hello World!";
strlen($str); // returns 12
?>
```



30

PHP String Functions

str_word_count()


The PHP str_word_count() function counts the number of words in a string

```
<?php
str_word_count("Hello World!"); // Returns 2
?>
```

Or

```
<?php
$str = "Hello World!";
str_word_oount($str); // returns 2
?>
```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

31

PHP String Functions

strpos()


The PHP strpos() function searches for a specific text within a string

If a match is found, the function returns the character position of the first match

If no match is found, it will return FALSE

```
<?php
echo strpos("Hello world!", "world"); // outputs 6
?>
```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

32

PHP String Functions


str_replace()

The PHP str_replace() function replaces some characters with some other char

acters in a string

```
<?php
echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!
?>
```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

33

PHP String Functions

explode ()

```

<?php
$sequence = "A,B,C,DE,F,G";
$elements = explode ("",$sequence);
// Now $elements is an array with all substrings between "," char
echo $elements[0]; // output: A;
echo $elements[1]; // output: B;
echo $elements[2]; // output: C;
echo $elements[3]; // output: DE;
echo $elements[5]; // output: F;
echo $elements[6]; // output: G;
?>

```



34

PHP String: " or `

. A single quoted strings is displayed "as-is"

```

$age = 37;
$str = 'I am $age years old'; // output: I am $age years old
$str = "I am $age years old"; // output: I am 37 years old

```



35

PHP User Defined Functions

PHP has over 1000 built-in functions that can be called directly, from within a script, to perform a specific task

Check: https://www.w3schools.com/php/php_ref_overview.asp



36

PHP User Defined Functions

Besides the built-in PHP functions, it is possible to create your own functions

- A function is a block of statements that can be used repeatedly in a program
- A function will not execute automatically when a page loads
- A function will be executed by a call to the function



37

PHP User Defined Functions

```
<?php
function functionName() {
    // code to be executed
}
?>
```

Note:A function name must start with a letter or an underscore. Function names are NOT case-sensitive



38

PHP User Defined Functions

```
<?php
function helloMessage() {
    echo "<h1>Hello World</h1>";
}

// later...
helloMessage(); // call the function
?>
```

```
<?php
function helloMessage() {
    echo "<h1>Hello World</h1>";
}
?>
<!-- blah, blah, HTML, blah -->
<?php
helloMessage(); // call the function
?>
```



39

PHP User Defined Functions

Information can be passed to functions through arguments. An argument is just like a variable

Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma

The following example has a function with one argument (**\$fname**). When the **familyName ()** function is called, we also pass along a name (e.g. Jani), and the name is used inside the function, which outputs several different first names, but an equal last name



40

PHP User Defined Functions

Keeping up with the Joneses

Casey Jones
Indiana Jones
John Paul Jones
Tommy Lee Jones
James Earl Jones



```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <h1>Keeping up with the Joneses</h1>
  <?php
    function familyName($fname) {
      echo "$fname Jones<br>";
    }

    familyName("Casey");
    familyName("Indiana");
    familyName("John Paul");
    familyName("Tommy Lee");
    familyName("James Earl");
  ?>
</body>
</html>
```



41

PHP Typing

PHP automatically associates a data type to the variable, depending on its value. Since the data types are not set in a strict sense, you can do things like adding a string to an integer without causing an error

In PHP 7, type declarations were added

Gives the option to specify the expected data type when declaring a function, and by adding the strict declaration, it will throw a "Fatal Error" if the data type mismatch



42

PHP Typing

```

<?php
function addNumbers(int $a, int $b) {
    return $a + $b;
}
echo addNumbers(5, "5 days");
// since 'strict' is not enabled,
// "5 days" is cast to int(5). This
// will return 10. Weird.
?>

```

43

PHP Typing

```

<?php
declare(strict_types = 1);

function addNumbers(int $a, int $b) {
    return $a + $b;
}
echo addNumbers(5, 5); // returns 10
echo addNumbers(5, "5 days");
// Fatal error: Uncaught TypeError...
?>
</body>

```

44

PHP Default Value

A default value can be set for a function when an argument isn't provided

45

PHP Default Value

```

<body>
  <?php
    declare(strict_types = 1);

    function setHeight(int $minHeight = 50) {
      echo "The height is : $minHeight <br>";
    }

    setHeight(250);
    setHeight(); // will use the default value
    setHeight(135);
  ?>
</body>

```



46

PHP Function Return

```

<body>
  <?php
    declare(strict_types = 1);

    function sum(int $x, int $y) {
      $z = $x + $y;
      return $z;
    }

    echo "5 + 10 = " . sum(5, 10) . "<br>";
  ?>
</body>

```



47

PHP Return Type Declarations

PHP 7 also supports Type Declarations for the return statement. Like with the type declaration for function arguments, by enabling the strict requirement, it will throw a "Fatal Error" on a type mismatch

To declare a type for the function return, add a colon (:) and the type right before the opening curly ({) bracket when declaring the function



48

PHP Return Type Declarations

```

<?php
declare(strict_types = 1);

function sum(float $a, float $b) : float {
    return $a + $b;
}

echo addNumbers(1.2, 5.3) . "<br>";
?>
</body>

```

49

PHP Superglobals

Some predefined variables in PHP are "superglobals"
 They are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special

50

PHP Superglobals

- \$GLOBALS → GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods)
- \$_SERVER → PHP super global variable which holds information about headers, paths, and script locations
- \$_REQUEST → We'll use these the most
- \$_POST
- \$_GET
- \$_FILES
- \$_ENV
- \$_COOKIE
- \$_SESSION

51

```

<!DOCTYPE html>
<html>
<head>...
</head>
<body>
  <h1>$_SERVER</h1>
  <?php
    echo "Server Type: " . $_SERVER['SERVER_SOFTWARE'] . "<br>";
    echo "Server Name: " . $_SERVER['SERVER_NAME'] . "<br>";
    echo "Server Host: " . $_SERVER['HTTP_HOST'] . "<br>";
    echo "Request From: " . $_SERVER['REMOTE_ADDR'] . "<br>";
    echo "Script Name: " . $_SERVER['SCRIPT_NAME'] . "<br>";
  ?>
</body>
</html>

```

\$_SERVER

Server Type: Apache/2.4.29 (Ubuntu)
 Server Name: ramseyjw.csci1720.net
 Server Host: ramseyjw.csci1720.net
 Request From: 97.89.32.238
 Script Name: /info.php

East Tennessee State University
Department of Computing
CSCI 1720
Intermediate Web Design

52

```

<!DOCTYPE html>
<html>
<head>...
</head>
<body>
  <form action=""><?php echo $_SERVER['PHP_SELF']; ?>>
    Name: <input type="text" name="fname">
    <input type="submit" value="Submit" />
  </form>
  <?php
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
      $name = $_POST["fname"];
      if (empty($name)) {
        echo "Name is empty";
      } else {
        echo $name;
      }
    }
  ?>

```

\$_REQUEST

Name: Fred

↓

Name:

East Tennessee State University
Department of Computing
CSCI 1720
Intermediate Web Design

53

PHP \$_GET & \$_POST

PHP **\$_GET** is a PHP super global variable which is used to collect form data after submitting an HTML form with method="get"

PHP **\$_POST** is a PHP super global variable which is used to collect form data after submitting an HTML form with method="post". **\$_POST** is also widely used to pass variables

East Tennessee State University
Department of Computing
CSCI 1720
Intermediate Web Design

54

PHP Form Handling [Review]

The PHP superglobals `$_GET` and `$_POST` are used to collect form-data

```
<form action="welcome.php" method="post">
  Name: <input type="text" name="name"><br>
  E-mail: <input type="text" name="email"><br>
  <input type="submit">
</form>
```



East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

55

PHP Form Handling [Review]

welcome.php

```
<?php
if(isset($_POST['name'])) {
    echo "Welcome " . $_POST['name'] . "<br>";
}
if (isset($_POST['email'])) {
    echo "The email you entered is: " . $_POST['email'];
}
?>
```

Welcome Jack
The email you entered is: ramseyjw@etsu.edu

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

56

PHP Form Handling [Review]

Both GET and POST create an array (e.g. array(key1=>value1, key2=>value2, key3=>value3, ...))


Holds key/value pairs, where keys are the names of the form controls and values are the input data from the user

! The 'key' value corresponds with the 'name' attribute in the form field

```
E-mail: <input type="text" name="email"><br>
```

The 'key' value for this field is "email"

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

57

PHP Form Handling

```

<!DOCTYPE html>
<html>
<head> ...
</head>
<body>
  <form action=""><php echo $_SERVER["PHP_SELF"]; ?>
    Name: <input type="text" name="name"><br>
    E-mail: <input type="text" name="email"><br>
    Position: <input type="text" name="pos"><br>
    Employer: <input type="text" name="emp"><br>
    <input type="submit" value="Submit" />
  </form>

```

Name: Jack Ramsey

E-mail: ramseyjw@etsu.edu

Position: Lecturer

Employer: ETSU

Submit

http://ramseyjw.csci1720.net/welcome.php?name=Jack+Ramsey&email=ramseyjw%40etsu.edu&position=Lecturer&employer=ETSU

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

58

PHP Form Handling


http://ramseyjw.csci1720.net/welcome.php?name=Jack+Ramsey&email=ramseyjw%40etsu.edu&position=Lecturer&employer=ETSU

```

<?php
// this is how the PHP interpreter "sees"
// the incoming $_GET superglobal (it's
// an associative array)
$_GET == [
    "name" => "Jack Ramsey",
    "email" => "ramseyjw@etsu.edu",
    "position" => "Lecturer",
    "employer" => "ETSU"];
?>

```

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

59

PHP Form Handling

Often, particularly with longer scripts, we'll assign each incoming argument to a local variable, for easier use

```


$name = $_GET["name"];
$email = $_GET["email"];
$pos = $_GET["pos"];
$emp = $_GET["emp"];

```

This way, we won't be stuck typing that awkward '\$_GET[""]' over and over again

⚠ NOTE: \$_POST works the same way, except the arguments are concealed in the request

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

60

PHP Form Handling


When to use GET?

Information sent from a form with the GET method is visible to everyone (all variable names and values are displayed in the URL)

GET also has limits on the amount of information to send. The limitation is about 2000 characters. However, because the variables are displayed in the URL, it is possible to bookmark the page. This can be useful in some cases

! Note: GET should NEVER be used for sending passwords or other sensitive information!

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

61

PHP Form Handling

```
firstName=Fred&lastName=Ferd&email=fred%40ferd.com&package=day2&tshirt=tshirt&tshirtQty=2&bagQty=1&thumbQty=1&numAttending=1
```


When to use POST?

Information sent from a form with the POST method is invisible to others (all names/values are embedded within the body of the HTTP request) and has no limits on the amount of information to send

However, because the variables are not displayed in the URL, it is not possible to bookmark the page

! Developers prefer POST for sending form data

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

62

PHP Form Handling

```
<form method="post" action="php/reg.php">
</form>
```

Here's an example of a form (we'll be making later)

The form's method is POST

So where is the data that's passed to the server?

Register Now!

First Name:

Last Name:

Email:

Registration Package

- Day 1 (\$40)
- Day 1 + Lunch (\$50)
- Day 1 + Day 2 (\$80)
- Day 1 + Day 2 + Lunch (\$100)
- Day 1-3 (\$140)
- Day 1-3 + Meals (\$140)

Enter


T-shirt (\$35)

Shoulder Bag (\$10)

Thumb Drive (\$24)

Attending

East Tennessee State University
Department of Computing



CSCI 1720
Web Design

63

PHP Form Handling

```

firstName=Fred&lastName=Ferd&email=fred%40ferd.com&package=day2&tshirt=tshirt&tshirtQty=2&bagQty=1&thumbQty=1&numAttending=1


```

- General
 - Request URL: http://localhost/csc1720.net/labs/registration/php/reg.php
 - Request Method: POST
 - Status Code: 200 OK
 - Remote Address: [::1]:80
 - Referrer Policy: no-referrer-when-downgrade
- Response Headers (7)
- Request Headers (17)
- Form Data [view parsed](#)

```

firstName=Fred&lastName=Ferd&email=fred%40ferd.com&package=day2&tshirt=tshirt&tshirtQty=2&bagQty=1&thumbQty=1&numAttending=1

```


East Tennessee State University Department of Computing  CSCI 1720 Intermediate Web Design

64

PHP Form Handling

One concern omitted from this example:

SECURITY

East Tennessee State University Department of Computing  CSCI 1720 Intermediate Web Design

65


PHP Objects

An object is a data type which stores data and information on how to process that data

In PHP, an object must be explicitly declared

First we must declare a class of object. For this, we use the class keyword

A class is a structure that can contain properties and methods:

East Tennessee State University Department of Computing  CSCI 1720 Intermediate Web Design

66

```

<?php
class Toon {
    public $name;
    public $level;

    function __construct($name, $level) {
        $this->name = $name;
        $this->level = $level;
    }

    function get_name() {
        return $this->name;
    }
}

// create an object
$cyrpus = new Toon("Cyprus", 75);

// show object properties
echo "Hunter's name: " . $cyrpus->name . "<br>";
echo $cyrpus->name . " is a level "
    . $cyrpus->level . " character";

```

Hunter's name: Cyprus
Cyprus is a level 75 character

We didn't use **get_name()** in this example, but it should be obvious how getters/ setters and other functions could be added

Note: class declaration does not include ()

CSCI 1720

Intermediate Web Design

67

Conclusion – For Now

Some folks argue that PHP is getting a little 'long in the tooth,' and is no longer relevant or at least is inferior to other options

Still the basis for a LOT of web applications (e.g., Wordpress)

Still evolving (hence, ver. 7.4.xxx)

One part of the stack

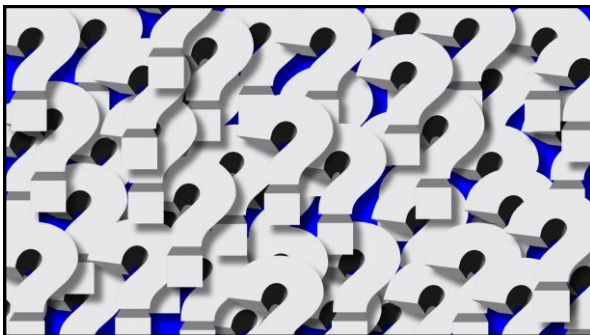
Frameworks – e.g., Laravel

East Tennessee State University
Department of Computing



CSCI 1720
Intermediate Web Design

68



69
