Server-side Scripting & PHP

9

Web Technology

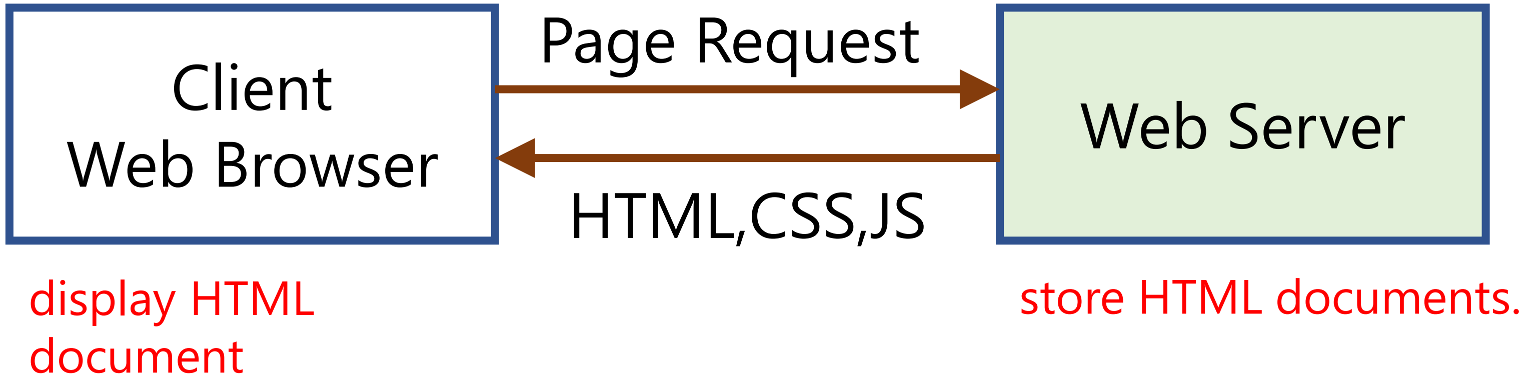
Asst. Prof. Manop Phankokkruad, Ph.D.
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang

Outline

- ❑ Server-side Scripting
- ❑ Introduction to PHP
- ❑ PHP Language basics
- ❑ PHP and the client

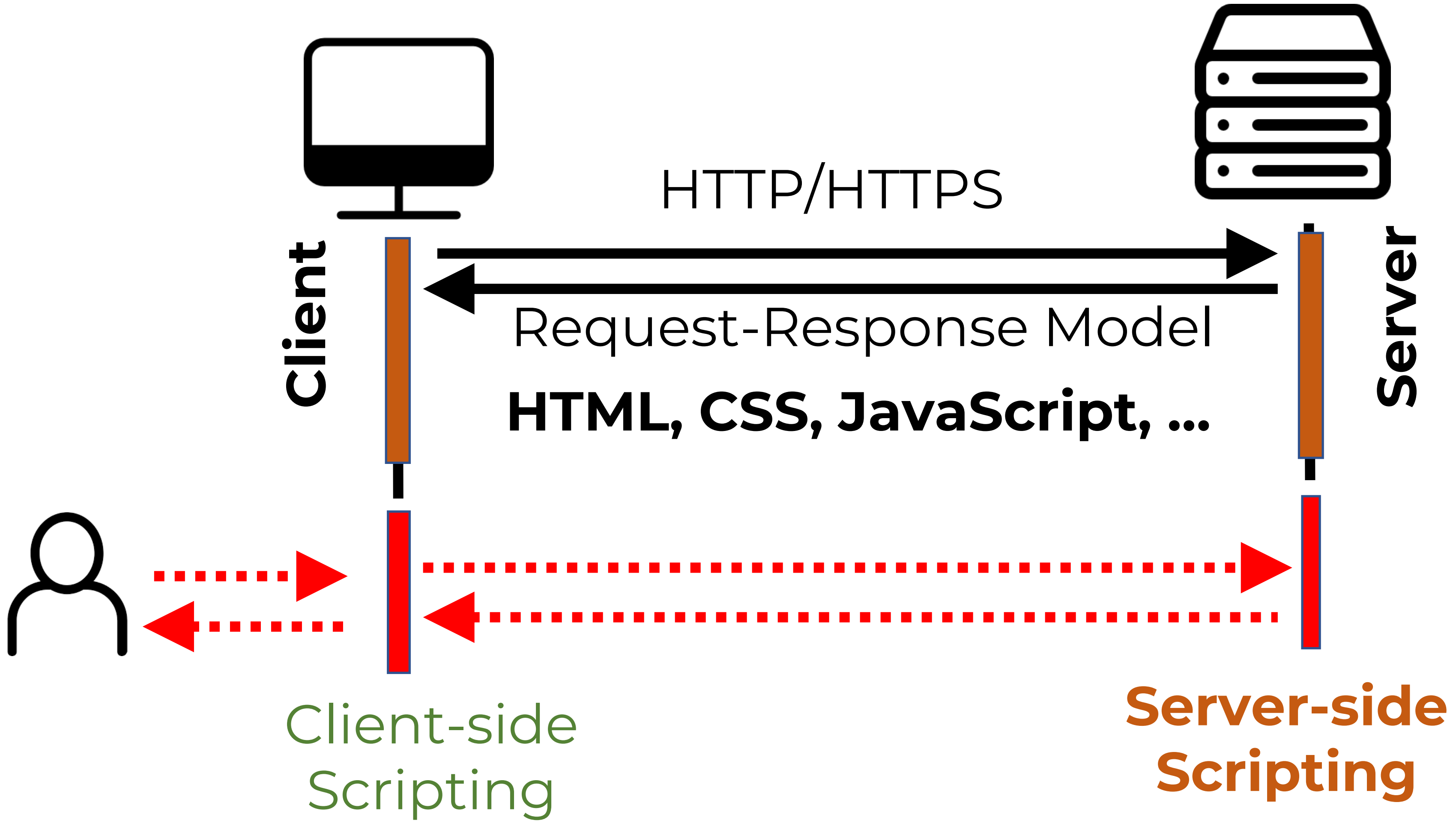
Introduction

- ❑ Standard web sites operate on a request/response basis.
- ❑ A user requests a resource E.g. HTML document.
- ❑ Server responds by delivering the document to the client.
- ❑ The client processes the document and displays it to user.



Introduction

Server-side technology



Server-side Scripting

“Server-side scripting is a technique used in web development which involves employing scripts on a web server which produce a response customized for each user’s request to the website.”

- ❑ Scripts can be written in any of a number of server-side scripting languages that are available.
- ❑ Server-side scripting is often used to provide a customized interface for the user.

Server-side Scripting

- ❑ Server-side scripting is distinguished from client-side scripting where embedded scripts, such as JavaScript, are run client-side in a web browser, but both techniques are often used together.
- ❑ **Server-side scripting** tends to be used for allowing users to have individual accounts and providing data from databases. It allows a level of privacy, personalization and provision of information that is very powerful.

Server-side Scripting

- ❑ PHP and ASP.net are the two main technologies for server-side scripting.
- ❑ The script is interpreted by the server meaning that it will always work the same way.
- ❑ Server-side scripts are never seen by the user. They run on the server and generate results which are sent to the user.
- ❑ Running all these scripts puts a lot of load onto a server but none on the user's system.

Server-side Scripting Languages

There are several server-side scripting languages available, including:

- ASP (*.asp)
- ASP.NET (*.aspx)
- Google Apps Script (*.gs)
- Java (*.jsp) via JavaServer Pages
- JavaScript using Server-side JavaScript (*.ssjs, *.js) such as Node.js
- Perl via the CGI.pm module (*.cgi, *.ipl, *.pl)
- PHP (*.php)
- Ruby (*.rb, *.rbw) such as Ruby on Rails

What is PHP?

❝ **PHP** is a server-side scripting language designed specifically for the Web. Within an HTML page, you can embed PHP code that will be executed each time the page is visited. ❞

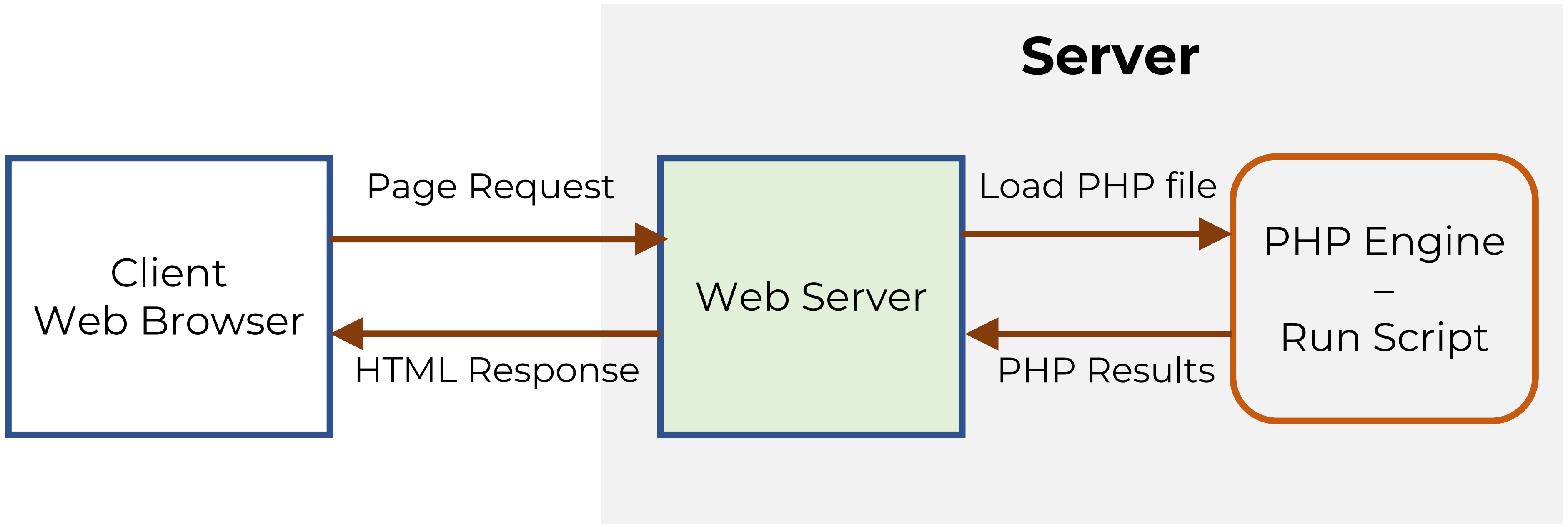
- ❑ PHP script is interpreted and executed on the server and generates HTML or other output.
- ❑ Multiple operating systems/web servers
- ❑ Execution is done before delivering content to the client.

What is PHP?

- ❑ Contains a vast library of functionality that programmers can handle.
- ❑ Executes entirely on the server, requiring no specific features from the client.
- ❑ Static resources such as regular HTML are simply output to the client from the server
- ❑ Dynamic resources such as PHP scripts are processed on the server prior to being output to the client

What is PHP?

❑ PHP has the capability of connecting to many database systems making the entire process transparent to the client



PHP Language Basics

The building blocks of the PHP language

- ❑ Syntax and structure
- ❑ Variables, constants and operators
- ❑ Data types and conversions
- ❑ Decision making IF and switch
- ❑ Interacting with the client application (HTML forms)

PHP - Syntax and Structure

- ❑ PHP is similar to C language
- ❑ All scripts start with **<?php** and with **?>**
- ❑ Line separator: **;** (semi-colon)
- ❑ Code block: **{** ..code here.. **}**
- ❑ White space is generally ignored (not in strings)
- ❑ Comments are created using:
  - ❑ // single line quote
  - ❑ /* Multiple line block quote */
- ❑ Precedence
  - ❑ Enforced using parentheses
  - ❑ $sum = 5 + 3 * 6;   // would equal 23
  - ❑ $sum = (5 + 3) * 6;   // would equal 48

PHP - Variables

- ❑ Prefixed with a **$**
- ❑ Assign values with = operator
- ❑ Example: $author = “Trevor Adams”;
- ❑ No need to define type
- ❑ Variable names are case sensitive
  - ❑ $author and $Author are different

PHP - Example Script

- ❑ PHP can be placed directly inside HTML

```
<html>
<head> <title>PHP Test</title> </head>
<body>
<?php
$author = “Trevor Adams”;
$msg = “Hello world!”;
echo $author .  “ says ” .  $msg;
?>
</body>
</html>
```

PHP - Constants

- Constants are special variables that cannot be changed. Use them for named items that will not change.
- Created using a define function
  - define('milestokm', 1.6);
  - Used without $
  - $km = 5 * milestokm;

```
<?php
   define('DEBUG',false);
   if (DEBUG) {      /* your code */    }
?>
```

PHP - Operators

- ❑ Standard mathematical operators
  - +, -, *, / and % (modulus)
- ❑ String concatenation with a period (.)
  - $car = “SEAT” . “ Altea”;
  - echo $car; would output “SEAT Altea”
- ❑ Basic Boolean comparison with “**==**”
  - Using only **=** will overwrite a variable value
  - Less than **<** and greater than **>**
  - **<=** and **>=** as above but include equality

PHP - Data Types

- ❑ PHP is **not** strictly typed, different to C and JAVA where all variables are declared
- ❑ A data type is either text or numeric. PHP decides what type a variable in an appropriate way automatically.
    - $vat_rate = 0.175;   // VAT Rate is numeric
    - echo $vat_rate * 100 . "%"; // outputs "17.5%"
    - $vat_rate is converted to a string for the purpose of the echo statement
- ❑ Object, Array and unknown also exist as types, Be aware of them but we shall not explore them today

Decision Making - Basics

- ❑ Decision making involves evaluating Boolean expressions (true / false)
- ❑ If ($catishungry) {   /* feed your cat */ }
- ❑ “**true**” and “**false**” are reserved words
- ❑ Initialise as **$valid = false**;
- ❑ Compare with **==**
- ❑ ‘and’, ‘&&’, ‘or’, ‘||’, ‘!’ (not) for combinations

***Example*:**
```
If ($catishungry && $havefood) {
    /* feed your cat*/
}
```

PHP - IF statement

- Used to perform a conditional branch

```
If (Boolean expression) {
// one or more commands if true

 } else {
// one or more commands if false

}
```

PHP - Switch Statements

- ❑ Useful when a Boolean expression may have many options E.g.

```
switch($choice) {
case 0: {  /* do things if choice equal 0 */  } break;
Case 1: {  /* do things if choice equal 1 */    } break;
Case 2: { /* do things if choice equal 2 */   } break;
Default: { /* do if choice is none of the above */ }
}
```

PHP - Switch Statements

❑ Switch statement example

```
$favcolor = "red";
switch ($favcolor) {
    case "red":
        echo "Your favorite color is red!"; break;
    case "blue":
        echo "Your favorite color is blue!"; break;
    default:
        echo "Your favorite color is neither red, nor blue!";
}
```

PHP - Arrays

- An array is a special variable, which can hold more than one value at a time.

```
$name = array();                    // create
$name = array(value0, value1, ..., valueN);
$name[index]          // get element value
$name[index] = value;  //  set element value
$name[] = value;                    // append
```

```
$a = array();                     // empty array (length 0)
$a[0] = 23;                       // stores 23 at index 0 (length 1)
$a2 = array("some", "strings", "in", "an", "array");
$a2[] = "Ooh!";          // add string to end (at index 5)
```

PHP - Associative Arrays

- ❑ Loop Through an Associative Array

  Associative arrays are “Arrays” that use named keys that you assign to them. There are two ways to create an associative array:

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
```

```
$age['Peter'] = "35";
$age['Ben'] = "37";
$age['Joe'] = "43";
```

```
foreach($age as $x => $x_value)  {
   echo "Key=" . $x . ", Value=" . $x_value . "<br>";
}
```

String compare functions

- ❑ Comparison can be partial matches and others.
- ❑ Variations with non case sensitive functions.

Name	Function
▪ strcmp()	Compare strings
▪ strstr)(), strchr()	find string/char within a string
▪ strpos()	find numerical position of string
▪ str_replace(), substr_replace()	replace string
▪ substr()	copy a part of string

String compare functions

- ❑ String comparison examples

```
// Replace "world" in "Hello world!" with "Peter"
echo str_replace("world","Peter","Hello world!");
```

```
$test = “Hello World! \n”;
print strpos($test, “o”);
print strpos($test, “o”, 5);
```

```
/* Find the first occurrence of "world" inside "Hello world!"
and return the rest of the string */
echo strstr("Hello world!","world");
```

PHP - Dealing with the Client

How is it useful in the web site?

- ❑ PHP allows developer to use HTML forms
- ❑ Forms require technology at the server to process them
- ❑ PHP is a feasible and good choice for the processing of HTML forms
- ❑ Implemented with a <form> element in HTML
- ❑ Contains other input, text area, list controls and options
- ❑ Has some method of submitting

PHP - Dealing with the Client

<form method=“post” action=“file.php” name=“frmid” >

- Method specifies how the data will be sent.
- Action specifies the file to go to “file.php”
- id gives the form a unique name

❑ **Post** method sends all contents of a form with basically hidden headers

PHP - Dealing with the client

- **Get** method sends all form input in the URL requested using name=value pairs separated by ampersands (&)
  - E.g. file.php?name=trevor&number=345
  - Is visible in the URL shown in the browser
- All form values are placed into an array
- “file.php” could access the form data using:
  - **$_POST[‘name’]**
- If the form used the **get** method, the form data would be available as:
  - **$_GET[‘name’]**

PHP - Dealing with the client

For example, an HTML form:

```
<form id="showmsg" action="show.php" method="post">
<input type="text" id="txtMsg" value="Hello World" />
<input type="submit" id="submit" value="Submit">
</form>
```

A file called show.php would receive the submitted data. It could output the message, for example:

```
<body>
<p> <?php echo $_POST["txtMsg"]; ?> </p>
</body>
```

More Information

- ❑ W3 Schools
  http://www.w3schools.com/php/
- ❑ PHP web site
  http://www.php.net/