University Mohamed Boudiaf - Msila
Faculty of Mathematics and Computer Science
Department of Computer Science
1st Year Master - RTIC

Web Technologies

Server-side languages

Mohamed Kamel

Academic Year 2023/2024

Table of Contents

Introduction

- PHP is a server scripting language
- Tool for making dynamic and interactive Web pages.
- Widely-used
- Free
- Powerful and efficient
- Cross platform
- Compatible with many servers (Apache, IIS, ...)
- Supports many databases

Syntax

- A PHP script starts with <?**php** and ends with ?>
- PHP statements end with a semicolon (;)
- PHP, keywords, classes, functions are not case-sensitive.

PHP example

```
<!DOCTYPE html>
<html>
<body>
  <?php
    EcHo "My first PHP script!";
  ?>
</body>
</html>
```

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**

PHP Capabilities

- 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
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data
- PHP can output images, PDFs, text and XML files

Comments, Variables

Example

```
// This is a single-line comment
$txt = "Hello world!";
$x = 5;
$y = 10.5;
```

Echo and Print

echo and **print** are more or less the same

- echo has no return value while print has a return value of 1
- echo can take multiple parameters
- echo is marginally faster than print

Example

```
echo "PHP example!<br>";
echo "Multiple ", "string ", "was ", "arguments";

$txt = "PHP";
$x = 5;
$y = 4;
echo "<h2>Learn " . $txt . "</h2>";
echo $x + $y;
```

Data Types

Example

```php
<?php
$x = "Hello world!"; // String
$x = 5985; // Integer
$x = null; // Null Value
$x = 10.365; // Float
$x = true; // Boolean
var_dump($x); // Returns the data type and value
?>
```

Data Types

Example

```
class Car {
  public $model;
  public function __construct($model) {
    $this->model = $model;
  }
  public function message() {
    return "My car is a " . $this->model . "!";
  }
}

$myCar = new Car("Renault");
echo $myCar -> message();
echo "<br>";
$myCar = new Car("Toyota");
echo $myCar -> message();
```

Strings

Example

```php
echo strlen("Hello world!"); // outputs 12
echo str_word_count("Hello world!"); // outputs 2
echo strrev("Hello world!"); // outputs !dlrow olleH
echo strpos("Hello world!", "world"); // outputs 6
echo str_replace("world", "Dolly", "Hello world!");
// outputs Hello Dolly!
```

Math

Example

```php
echo(pi()); // returns 3.1415926535898
echo(min(0, 150, 30, 20, -8, -200)); // returns -200
echo(max(0, 150, 30, 20, -8, -200)); // returns 150
echo(abs(-6.7)); // returns 6.7
echo(sqrt(64)); // returns 8
echo(sqrt(64)); // returns 8
echo(rand()); // Random number
echo(rand(10, 100)); // Random number between 10 and 100
```

Constants

Example

```php
define("MI", "Mathematiques et Informatique");
echo MI; // prints Mathematiques et Informatique

define("names", ["Ahmed", "Ibrahim", "Ali"]);
echo names[0]; // prints Ahmed
```

Operations

Example

```php
var_dump(50 == "50"); // prints bool(true)
var_dump(50 === "50"); // prints bool(false)

echo 10<=>20; // prints -1
echo 10<=>10; // prints 0
echo 10<=>5; // prints 1

echo "Mathematiques" . " et " . "Informatique";
  // prints Mathematiques et Informatique

echo 10 === "10"?"YES":"NO"; //prints
```

switch

In PHP, switch statement can be used with strings too

Example

```php
<?php
$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";
}
?>
```

Loops

Example

```php
$i = 0;
while($i < 5) {
  echo "$i "; $i++; // prints 0 1 2 3 4
}

do {
  echo "$i "; $i--; // prints 5 4 3 2 1 0
} while ($i >= 0);

for ($i = 0; $i < 50; $i+=10)
echo "$i "; // prints 0 10 20 30 40

$colors = array("red", "green", "blue", "yellow");
foreach ($colors as $value)
  echo "$value "; // prints red green blue yellow
```

Functions

Example

```
function myFunction($word1, $word2) {
  echo "$word1 $word2";
}
myFunction("Hello", "world!"); // prints Hello world!
```

Arrays

Indexed arrays

```
$cars = array("Renault", "BMW", "Peugeot");
echo "I like " . $cars[0] . ".";
```

Associative arrays

```
$age = array("Ahmed"=>"35", "Ibrahim"=>"37", "Oussama"=>"43");
echo "Oussama is " . $age['Oussama'] . " years old.";
```

Multidimensional arrays

```
$cars = array ( array("Renault",22,18), array("BMW",15,13));
echo $cars[0][0];
```

Sorting arrays

- **sort()**, sort arrays in ascending order
- **rsort()**, sort arrays in descending order
- **asort()**, sort associative arrays in ascending order, according to the value
- **ksort()**, sort associative arrays in ascending order, according to the key
- **arsort()**, sort associative arrays in descending order, according to the value
- **krsort()**, sort associative arrays in descending order, according to the key

Example

```php
$cars = array("Mohamed", "Ali", "Abderrahamane");
sort($cars);
foreach($cars as $car) echo $car, " ";
// prints Abderrahamane Ali Mohamed
```

Superglobals

- $GLOBALS
- $_SERVER
- $_REQUEST
- $_POST
- $_GET
- $_FILES
- $_ENV
- $_COOKIE
- $_SESSION

$GLOBALS

$GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script

Example

```
$x = 75;
$y = 25;

function addition() {
  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}

addition();
echo $z;
```

$_SERVER

$_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

Example

```
echo $_SERVER['SERVER_NAME'], "<br>";
echo $_SERVER['REQUEST_METHOD'], "<br>";
echo $_SERVER['HTTP_USER_AGENT'], "<br>";
echo $_SERVER['PHP_SELF'];
```

$_REQUEST

Example

```
<form method="post"
action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // collect value of input field
  $name = $_REQUEST['fname'];
  if (empty($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>
```

$_REQUEST & $_POST

Example

```
<form method="post"
action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // collect value of input field
  $name = $_REQUEST['fname']; // Or $_POST
  if (empty($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>
```

$_GET

Link

```
<a href="test.php?subject=TechWeb&dep=info">Test $GET</a>
```

Processing

```
<?php
echo "Study " . $_GET['subject'] . " at " . $_GET['dep'];
?>
```

RegEx

A regular expression is a sequence of characters that forms a search pattern. When you search for data in a text, you can use this search pattern to describe what you are searching for.

- **preg_match()**, Returns 1 if the pattern was found in the string and 0 if not
- **preg_match_all()**, Returns the number of times the pattern was found in the string, which may also be 0
- **preg_replace()**, Returns a new string where matched patterns have been replaced with another string

RegEx

Example

```php
<?php
$str = "abcdef abcdef";
$pattern = "/cde/";
echo preg_match($pattern, $str); // Outputs 1
echo preg_match_all($pattern, $str); // Outputs 2

?>
```

RegEx

- Modifiers
  - i, case-insensitive search
  - m, multiline search
  - u, UTF-8 search
- Brackets
  - [*abc*] Find one character from the options between the brackets
  - [*bc*] Find any character NOT between the brackets
  - [0 – 9] Find one character from the range 0 to 9
- Metacharacters
  - | Or, in: cat|dog|fish
  - . One instance of any character
  - ^Finds a match as the beginning of a string as in: ^Hello
  - $ Finds a match at the end of the string as in: World$
  - *d* Finds a digit
  - *s* Finds a whitespace character
  - *b* Finds a match at the beginning of a word like this: *b*WORD, or at the end of a word like this: WORD*b*
  - *uxxxx* Finds the Unicode character specified by the hexadecimal number xxxx

Include

- **include** and **require** are used to insert the content of one PHP file into another.
- **require** produces fatal error upon failure where **include** produces warning.

footer.php

```
<?php
  echo "<p>Copyright 2022</p>";
?>
```

mypage.php

```
<p>Some text.</p>
<?php include 'footer.php';?>
```

Include

- **include_once** and **require_once** are similar to **include** and **require** with the only difference being that the file will be included just once.

mypage.php

```
<?php include_once 'footer.php';?>
<?php include_once 'footer.php';?>
```

Cookies

- Cookies are used to identify a user.
- A cookie is a small data that the server sends to the browser, each time the same user requests a page the browser will send that cookie to the server
- The **setcookie()** function must appear BEFORE the <**html**> tag.

Cookie example

```
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30),
   "/"); // 86400 = 1 day
?>
```

Sessions

- A session is a way to store variable in the server across pages
- Session variables are not stored in user's computer
- A session is started with the **session_start()** function.
- The **session_start()** function must appear BEFORE the <**html**> tag.

Session example

```
<?php session_start();?>
<html>
...
<?php
$_SESSION["username"] = "oussama123";
?>
...
```

JSON

- PHP has some built-in functions to handle JSON.
- **json_encode()** is used to encode PHP data into JSON format
- **json_decode()** is used to decode JSON data into PHP data

JSON encode example

```php
<?php
$marks = array("Math"=>15, "Phys"=>13, "Sci"=>16);
echo json_encode($marks);
?>
```

JSON decode example

```php
<?php
$json_object = '{"Math":15,"Phys":13,"Sci":16}';
echo json_decode($json_object);
?>
```

Database

In this module we will use MySQL Database.

- MySQL is the most popular database system used with PHP
- MySQL is a database system used on the web
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, and easy to use
- MySQL compiles on a number of platforms
- MySQL is free to download and use
- MySQL is developed, distributed, and supported by Oracle Corporation

Connecting to Database

PHP 5 and later can work with a MySQL database using:

- MySQLi extension (the "i" stands for improved), MySQLi will only work with MySQL
- PDO (PHP Data Objects), PDO will work with other database systems

Connecting to Database system

MySQLi Object-Oriented

```php
<?php
$c = new mysqli("localhost", "root", "");
if ($c->connect_error) {
  die("Connection failed: " . $co->connect_error);
}
echo "Connected successfully";
// Working with database
$c->close();
?>
```

Connecting to Database

MySQLi procedural

```php
<?php
$c = mysqli_connect("localhost", "root", "");
if (!$c) {
  die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
// Working with database
mysqli_close($c);
?>
```

Connecting to Database

PDO

```
try {
  $c = new PDO("mysql:host=localhost;dbname=myDB",
    "username", "password");
  $c->setAttribute(PDO::ATTR_ERRMODE,
    PDO::ERRMODE_EXCEPTION);
  echo "Connected successfully";
  // Working with database
  $c = null;
} catch(PDOException $e) {
  echo "Connection failed: " . $e->getMessage();
}
```

Creating Database

PDO

```
try {
  $c = new PDO("mysql:host=localhost;dbname=myDB",
    "username", "password");
  $c->setAttribute(PDO::ATTR_ERRMODE,
    PDO::ERRMODE_EXCEPTION);
  echo "Connected successfully";
  // Working with database
  $c = null;
} catch(PDOException $e) {
  echo "Connection failed: " . $e->getMessage();
}
```

Questions ?