



Cours Framework de Développement Web

Nahla Haddar
Université de Sfax
Membre du laboratoire MIRACL

Audience: D-LSI-ADBD
Année-universitaire: 2023-2024

2

Avant-propos

- Dans ce cours vous allez apprendre à développer des scripts PHP native et PHP orienté objets, puis à une stade avancée, vous allez apprendre à développer des applications web en utilisant le Framework **Laravel**.
- Avec Laravel, vous apprendrez à:
 - intégrer des vues avec le moteur de gabarits **Blade**,
 - manipuler une base de données à l'aide de l'**ORM** (Object-relational mapping) **Eloquent**
 - interagir avec vos utilisateurs à l'aide de **formulaires parfaitement intégrés et validés**.

3

Plan du cours

- Initiation au PHP native et PHP orienté objets
- Introduction et installation du framework Laravel,
- Principe de fonctionnement selon le modèle MVC (Model-View-Controller)
- Modes de routage et paramètres de substitution
- Création de contrôleur (Controller)
- Création de templates (View) avec Blade ,
- Les formulaires: création et validation
- Manipulation de BD avec Eloquent ,
- Création d'application CRUD (Create, Read, Update, Delete)

4

Introduction

Sites Web

- Un site Web est un ensemble de pages Web stockés dans un serveur Web (tels que Apache).
- Il existe deux types de sites Web:
 - Les sites statiques
 - Les sites dynamiques

Les sites statiques

- Sont réalisés uniquement à l'aide des langages **HTML**, **CSS** et **JavaScript** (site vitrine),
- Leur contenu ne peut pas être changé que par l'intervention du webmaster.

Les sites dynamiques

- Utilisent HTML + CSS + JavaScript + des langages web dynamique (PHP, JSP/Servlet, Python...).
- Leur contenu est dit « dynamique » parce qu'il est stocké dans une **base de données** et il peut changer sans l'intervention du webmaster.
- Cette base de données doit être obligatoirement située dans un **serveur de base de données** (tels que MySql, SQLServer, etc.).

Client ... Serveur

- Un **Serveur** est une application située sur un ordinateur très puissant, capable de gérer un grand nombre de requêtes simultanément.
- Un **Client** est une application qui se connecte à un serveur web pour obtenir ou modifier des informations à l'aide de requêtes (par exemple le navigateur web).

9

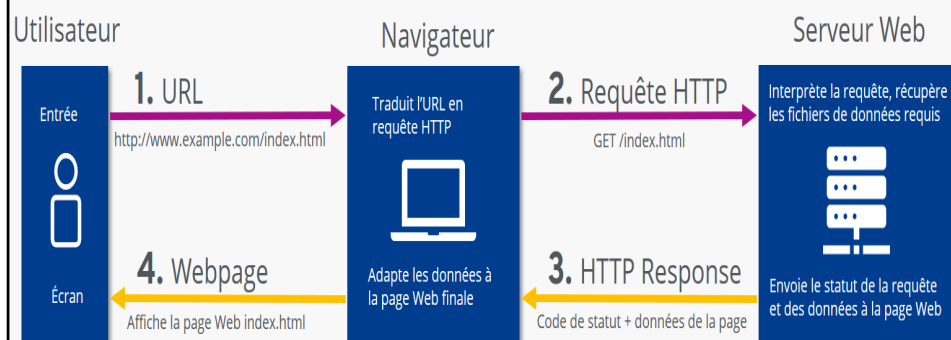
Le protocole HTTP

- HTTP (Hypertext Transfer Protocol) est un protocole de communication entre un client et un serveur.
- Le client demande une page (ressource) au serveur en envoyant une requête et le serveur réagit en envoyant une réponse, qui est en général une page Html.
- Quand on surfe sur Internet chacun de nos clics provoque en général cet échange.



10

Processus de communication HTTP

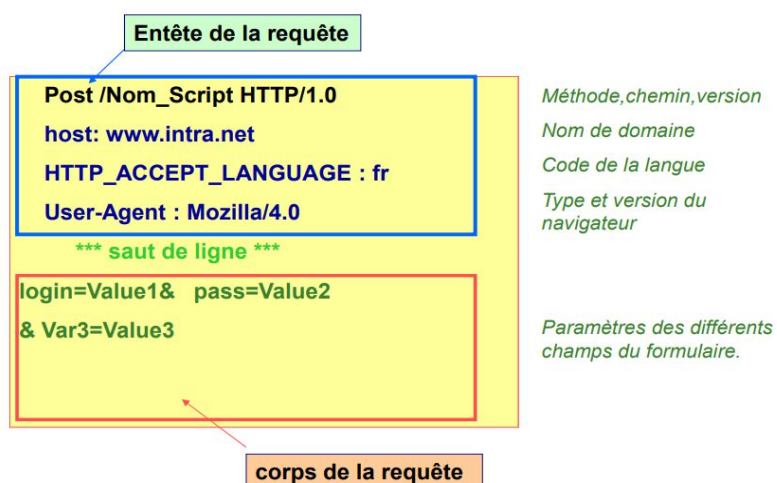


Dialogue HTTP

- Deux types de dialogue:
 - **Récupération d'un document** (par le clic sur un lien ou l'écriture de son URL dans la barre d'adresse du navigateur)
 - méthode **GET**
 - **Soumission d'un formulaire**
 - méthodes **GET** ou **POST**

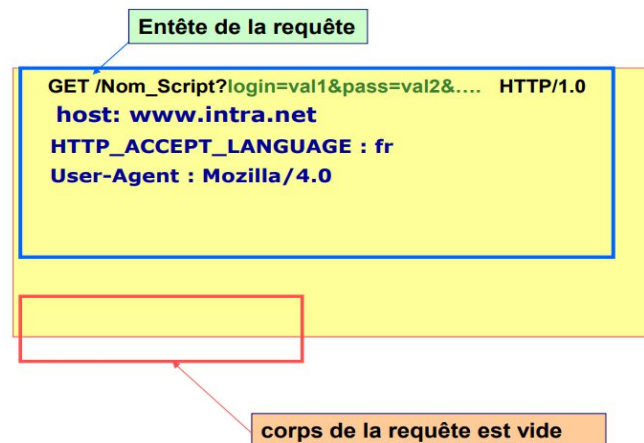
12

Requête HTTP de type POST

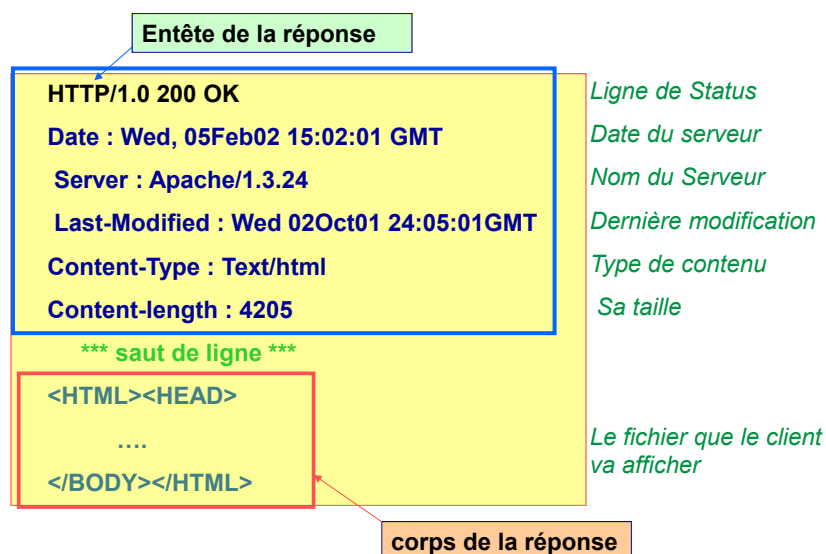


13

Requête HTTP de type GET



Réponse du serveur



Réponse du serveur

- De très nombreux statuts existent, parmi les plus connus :
 - **200**, la page a été retournée sans erreur du serveur ;
 - **404**, le code HTTP pour une ressource qui n'a pas été retrouvée sur le serveur ;
 - les codes **3XX**, qui signalent les redirections de ressources ;
 - les codes **4XX**, qui signalent une erreur côté utilisateur/client ;
 - les codes **5XX**, qui signalent une erreur côté serveur.

PHP : Les bases du langage

17

Caractéristiques Principales

- **PHP:** Signifie d'abord Personal Home Pages puis HypertextPreProcessor
- **Langage interprété**
 - Son interpréteur se nomme **Zend Engine**
 - Pas de compilation
 - Exécute instruction par instruction
 - Multi-plateformes
- Spécialisé dans la génération de texte ou de documents
 - HTML, PDF, Images
- **Fichiers d'extension .php**
 - Code PHP+ balises HTML

18

Imbrication de code HTML et PHP

- Une page PHP peut être entièrement programmée en PHP ou mélangée avec du code html.
- Pour différencier le code PHP des balises HTML, on utilise des balises particulières:
 - `<?php ... ?>` (forme préférée)
 - `<script language="php">... </script>`

19

Exemple simple

Script.php

```
<!DOCTYPE html>
<html>
<body>

<?php
echo "My first PHP script!";
?>

</body>
</html>
```

Résultat:

My first PHP script!

Comment tester notre code ?

20

Installation de l'environnement de travail

- Deux solutions sont possibles:

- 1^{ère} solution:

- Installer un environnement web en local tel que **XAMPP**, **EASYPHP**, **CADDY**,... Cela vous permettra d'avoir **PHP**, **Apache** et **MySQL** installés en local

- 2^{ème} solution (préférable):

- Installer **Laragon**: <https://laragon.org/download/index.html> un environnement de développement web assez complet:
 - Offre un **serveur Apache**, **serveur de BD (MySQL, PostgreSQL, NoSQL, des logiciels de gestion de BD (PhpMyAdmin, RoboMongo, PgAdmin), Composer**, ... et encore beaucoup plus.

21

Installation de Laragon

Why Laragon? Testimonials Download About Community

Download

Laragon is a universal development environment. It has many features to make you more productive:

[Benefits of Laragon](#)

After downloading, You can add [git](#), [phpmyadmin](#), [Node.js/MongoDB](#), [Python/Django/Flask/Postgres](#), [Ruby](#), [Java](#), [Go](#) using "[Tools > Quick add](#)"

Note: [You can also download from GitHub](#)

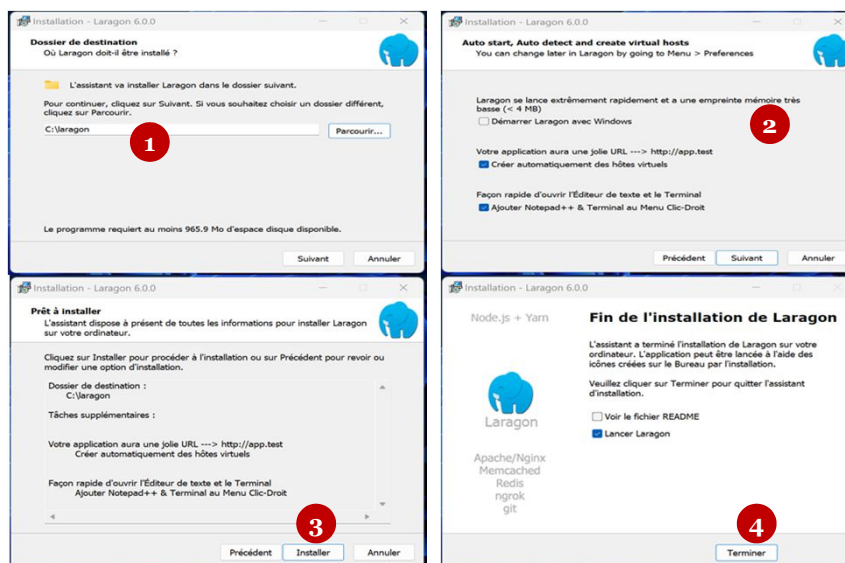
Edition

[Download Laragon - Full \(173 MB\)](#)

- **Laragon Full (64-bit):** Apache 2.4, Nginx, MySQL 8, PHP 8, Redis, Memcached, Node.js 18, npm, git

22

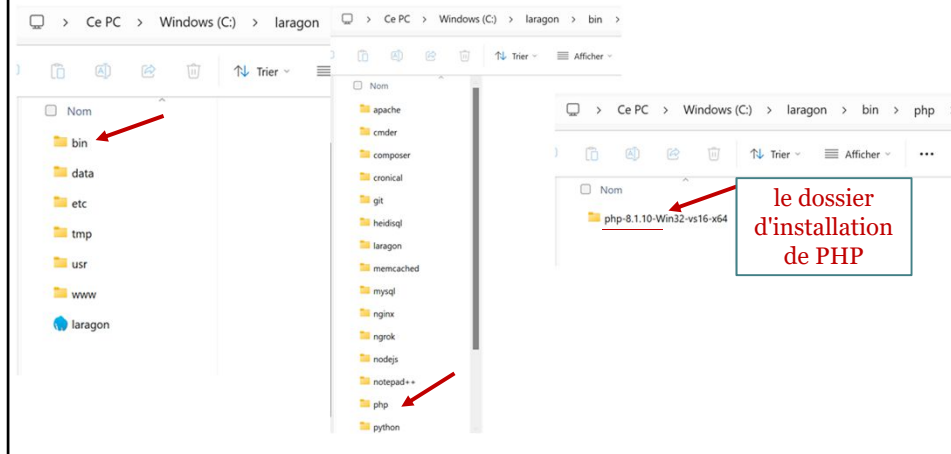
Installation de Laragon



23

Installation de Laragon

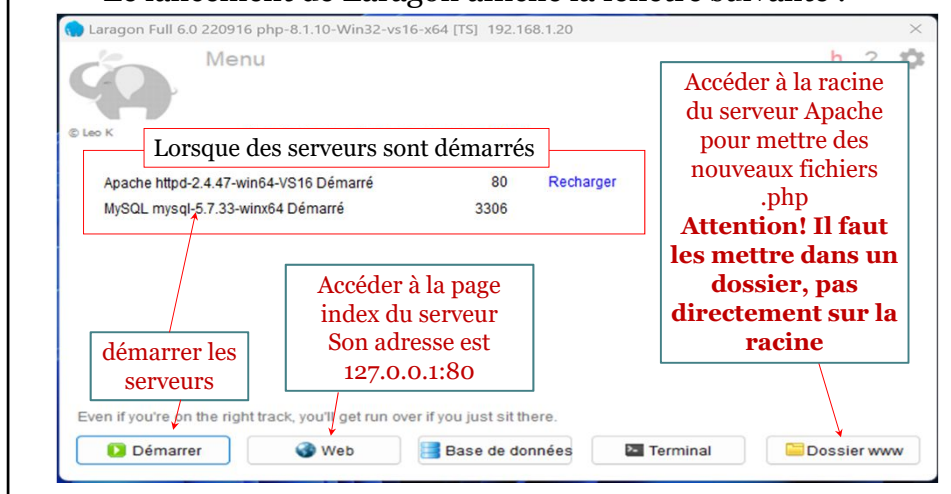
- l'installation [1-4] donne naissance à l'arborescence suivante :



24

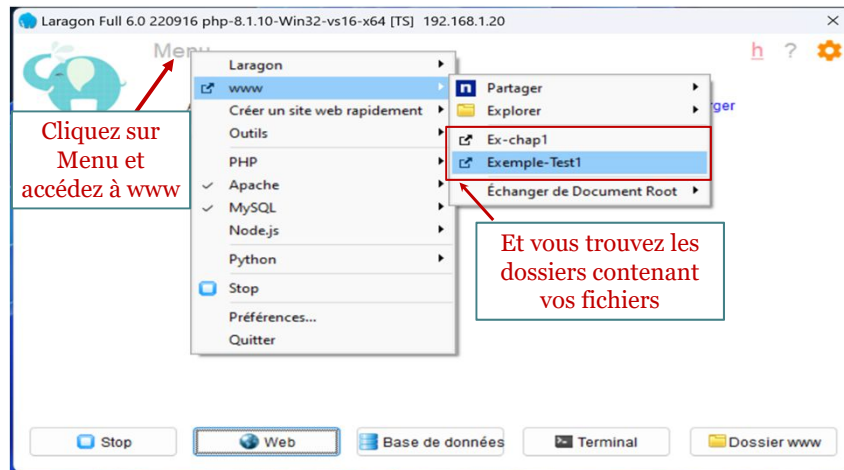
Prise en main de Laragon

- Le lancement de Laragon affiche la fenêtre suivante :



25

Comment tester vos fichiers « .php » ?



26

Comment tester vos fichiers « .php » ?

- Revenons à notre premier exemple



27

Les commentaires

- Les commentaires s'utilisent comme en C et en C++ avec :
 - `/* commentaire sur plusieurs ligne*/`
 - et `//` ou `#` `commentaire sur une seule ligne`
- **Exemple:**

```
<!DOCTYPE html>
<html>
<body>

<?php
// This is a single-line comment

# This is also a single-line comment
?>

</body>
</html>
```

28

Les commentaires

- Les commentaires s'utilisent comme en C et en C++ avec :
 - `/* commentaire sur plusieurs ligne*/`
 - et `//` ou `#` `commentaire sur une seule ligne`
- **Exemple:**

```
<!DOCTYPE html>
<html>
<body>

<?php
// This is a single-line comment

# This is also a single-line comment
?>

</body>
</html>
```

29

Affichage sur écran

- `echo` et `print` sont plus ou moins les mêmes.
- Ils sont tous deux utilisés pour afficher des données à l'écran.
- `echo` n'a pas de valeur de retour alors que `print` a une valeur de retour de 1
- `echo` peut avoir plusieurs paramètres alors que `print` non
- `echo` est légèrement plus rapide que `print`

30

Affichage sur écran

- **Affichage de texte (peut contenir des balises HTML)**

```
<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
?>
```

PHP is Fun!

Hello world!
I'm about to learn PHP!
This string was made with multiple parameters.

```
<?php
print "<h2>PHP is Fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

PHP is Fun!

Hello world!
I'm about to learn PHP!

31

Les variables en PHP:

1. Déclaration des variables

- Une variable commence par un dollar \$ suivi d'un nom de variable.
- Les variables ne sont pas typées au moment de leur création.

```
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
?>
```

- Attention PHP est sensible à la casse : var et Var ne sont pas les mêmes variables !
- **Les règles à respecter :**
 - Une variable peut commencer par une lettre
 - Une variable peut commencer par un souligné (underscore) « _ »
 - Une variable ne doit pas commencer par un chiffre.

32

Les variables en PHP:

2. Affichage des variables

- Avec `echo` ou `print` même syntaxe:
 - Mais ... attention au **guillemets** "..." ou **quotes** '...'

```
<?php
$txt1 = "Learn PHP";
$txt2 = "ISIMS";
$x = 5;
$y = 4;
```

Avec **les quotes** la **concaténation** entre les variables et le texte **est toujours obligatoire**.

```
echo "<h2> $txt1 </h2>";
echo 'Study PHP at ' . $txt2 . '<br>';
echo "$x + $y = " . ($x + $y);
?>
```

Avec **les guillemets** la **concaténation n'est obligatoire que en cas d'évaluation d'une expression**.

Learn PHP

Study PHP at ISIMS
5 + 4 = 9

33

Les variables en PHP:

3. Les constantes

- Les constantes sont automatiquement globales et peuvent être utilisées dans l'ensemble du script.
- **Syntaxe:**
 - `define("nom_constante", valeur_constante)`
- **Exemples:**

Une chaîne de caractère

```
<?php
define("GREETING", "Welcome to W3Schools.com!");
?>
```

Un tableau de chaînes de caractères

```
<?php
define("cars", [
    "Alfa Romeo",
    "BMW",
    "Toyota"
]);
?>
```

34

Les variables en PHP:

4. Portée des variables

- PHP a trois portées de variables différentes :
 - Locale
 - Globale
 - Statique

35

Les variables en PHP:

4. Portée des variables (LOCALE)

- Une variable déclarée dans une fonction a une **PORTÉE LOCALE** et n'est accessible qu'au sein de cette fonction :

```
<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>
```

36

Les variables en PHP:

4. Portée des variables (GLOBALE)

- Une variable déclarée en dehors d'une fonction a une **portée GLOBALE** et **n'est accessible qu'en dehors d'une fonction** :

```
<?php
$x = 5; // global scope

function myTest() {
    // using x inside this function will generate an error
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

echo "<p>Variable x outside function is: $x</p>";
?>
```

**Mais, normalement une variable globale est accessible dans les fonctions !
Comment doit-on faire alors ?**

37

Les variables en PHP:

4. Portée des variables (GLOBALE)

- PHP stocke également toutes les variables globales dans un tableau appelé `$GLOBALS['nom_var']`.
- Ce tableau est également accessible depuis les fonctions et peut être utilisé pour mettre à jour directement les variables globales.

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

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

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

38

Les variables en PHP:

4. Portée des variables (STATIQUE)

- Lorsqu'une **fonction est terminée/exécutée**, toutes ses **variables sont supprimées**.
- Cependant, nous voulons parfois qu'une variable locale **ne soit PAS supprimée**.
- Pour cela, utilisez le mot-clé **static** lors de la première déclaration de la variable :

```
<!DOCTYPE html>
<html>
<body>

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

myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
?>

</body>
</html>
```

0
1
2


39

Les variables en PHP:

5. Fonctions sur des variables

- Vérifier l'existence d'une variable (isset)

```
<?php
$a = "une variable en PHP";
if(isset($a)) echo "la variable a existe";
unset($a);
echo "la variable a a été supprimée ...";
```

- Tester si une variable est vide (empty)

```
<?php
$a = "une variable en PHP";
if (!empty($a)) echo " La variable existe et elle n'est pas vide !";
```

- **Attention!** La fonction empty() répond vrai si la variable n'existe pas et ceci sans faire aucun warning !

40

Les chaînes en PHP:

1. Les bases

- La chaîne est traitée comme un tableau de caractères indexé par un entier => \$str="Hello" ; echo \$str[1]; \\affiche 'e'
- **La concaténation à l'aide de .**

```
$str="Salut les Amis !\n";
$str.="Comment ça va ?"; // "Salut les Amis !\nComment ça va ?
$str2=$str."\n"; // "Salut les Amis !\nComment ça va ?\n"
```

- **La longueur d'une chaîne :**

```
<?php
echo strlen("Hello world!"); // outputs 12
?>
```

- **Compter le nombre de mots d'une chaîne**

```
<?php
echo str_word_count("Hello world!"); // outputs 2
?>
```

41

Les chaînes en PHP:

2. Les fonctions

- **Mettre en majuscules/minuscules :**
 - avec **strtoupper()** pour obtenir des majuscules
 - avec **strtolower()** pour mettre en minuscules
 - avec **ucfirst()** pour mettre en majuscule la première lettre d'une chaîne
 - avec **ucwords()** pour mettre en majuscule la première lettre de chaque mot dans une chaîne
- **Exemple:**

```
<?php
$str = "Marie A un Petit Agneau, et l'aime TRÈS fORT.";
$str = strtolower($str);
echo $str; // marie a un petit agneau, et l'aime très fort.
?>
```

42

Les chaînes en PHP:

2. Les fonctions

- **Remplace le texte dans une chaîne:**

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

- **Inverser une chaîne:**

```
<?php
echo strrev("Hello world!"); // outputs !dlrow olleH
?>
```

- **Vérifier si une chaîne est numérique:**

```
$x = "5985";
echo is_numeric($x); // affiche True
```

43

Les chaînes en PHP:

3. Recherche de sous-chaînes

- **strstr()**

- Il recherche la première occurrence d'une chaîne à l'intérieur d'une autre chaîne et affiche la portion de cette dernière à partir de la première occurrence rencontrée.
- Si la chaîne n'existe pas, elle strstr() **retourne faux**
- Cette fonction est insensible à la casse.

```
<?php
$email = 'USER@EXAMPLE.com';
echo strstr($email, 'e'); // Affiche ER@EXAMPLE.com
echo strstr($email, 'e', true); // Depuis PHP 5.3.0, Affiche US
?>
```

- **stristr()** fait le même travail mais elle est sensible à la casse.

44

Les chaînes en PHP:

Activité

- Transformez une chaîne écrite dans des casses différentes afin que chaque mot ait une initiale en majuscule.

- **Exemple:**

```
$ch="TransFormeZ unE Chaîne éCRITe dans des cASses diFFérenTes afiN
qUe chAQue MOT ait une inITiale en MAJUSCULE";
```

- **Résultat:**

```
Transformez Une Chaîne Écrite Dans Des Casses Différentes Afin Que
Chaque Mot Ait Une Initiale En Majuscule
```

- **Solution:**

45

Les tableaux en PHP

1. Tableaux simples

- Un tableau stocke plusieurs valeurs dans une seule variable

- **Exemple:**

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".<br/>";
print_r($cars);# affiche le contenu d'un tableau
?>
```

- Les **indices** des cases sont des entiers allons de **0** à **count(\$cars)-1**

- **Résultat:**

```
I like Volvo, BMW and Toyota.
Array ( [0] => Volvo [1] => BMW [2] => Toyota )
```

46

Les tableaux en PHP

2. Tableaux associatifs

- Les **indices** des cases sont **des chaînes de caractères**
=> **les indices sont des clés nommées**
- Il existe deux manières de créer un tableau associatif :

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

- Ou

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

- **Exemple qui affiche:** Peter is 35 years old.

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

47

Les tableaux en PHP:

3. Trie des tableaux

- `sort($tab)` - trie un tableau simple \$tab par ordre croissant
- `rsort($tab)` - trie un tableau simple \$tab par ordre décroissant
- `asort($tab)` - trie un tableau associatif \$tab par ordre croissant, selon la valeur
- `ksort($tab)` - trie un tableau associatif \$tab par ordre croissant, selon la clé
- `arsort()` - trie un tableau associatif \$tab par ordre décroissant, selon la valeur
- `krsort()` - trie un tableau associatif \$tab par ordre décroissant, selon la clé

48

Les tableaux en PHP:

3. Trie des tableaux

• Exemple:

```
<body>
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
arsort($age);
print_r($age) # Affiche des informations lisibles pour une variable
?>
</body>
```

Exécution:

```
Array ( [Joe] => 43 [Ben] => 37 [Peter] => 35 )
```


49

Les opérateurs en PHP

1. Opérateur arithmétique

Opérateur	Nom	Exemple
+	addition	$\$x + \y
-	Soustraction	$\$x - \y
*	Multiplication	$\$x * \y
/	Division	$\$x / \y
%	Module (reste de la division)	$\$x \% \y
**	Exponentiation	$\$x ** \y

50

Les opérateurs en PHP

2. Opérateur d'affectation

Affectation	Équivalent à ...
$\$x = \y	$\$x = \y
$\$x += \y	$\$x = \$x + \$y$
$\$x -= \y	$\$x = \$x - \$y$
$\$x *= \y	$\$x = \$x * \$y$
$\$x /= \y	$\$x = \$x / \$y$
$\$x \% = \y	$\$x = \$x \% \$y$

51

Les opérateurs en PHP

3. Opérateur de comparaison

Opérateur	Exemple	Résultat
==	<code>\$x == \$y</code>	Renvoie true si \$x est égal à \$y
===	<code>\$x === \$y</code>	Renvoie true si \$x est égal à \$y et qu'ils sont du même type
!=	<code>\$x != \$y</code>	Renvoie true si \$x n'est pas égal à \$y
<>	<code>\$x <> \$y</code>	Renvoie true si \$x n'est pas égal à \$y
!==	<code>\$x !== \$y</code>	Renvoie true si \$x n'est pas égal à \$y, ou s'ils ne sont pas du même type
>	<code>\$x > \$y</code>	Renvoie true si \$x est supérieur à \$y
<	<code>\$x < \$y</code>	Renvoie true si \$x est inférieur à \$y
>=	<code>\$x >= \$y</code>	Renvoie true si \$x est supérieur ou égal à \$y
<=	<code>\$x <= \$y</code>	Renvoie true si \$x est inférieur ou égal à \$y

52

Les opérateurs en PHP

3. Opérateur logique

Opérateur	Exemple	Résultat
and	<code>\$x and \$y</code>	True si \$x et \$y sont vrais
or	<code>\$x or \$y</code>	True si \$x ou \$y est vrai, ou les deux
xor	<code>\$x xor \$y</code>	True si \$x ou \$y est vrai, mais pas les deux
&&	<code>\$x && \$y</code>	True si \$x et \$y sont vrais
	<code>\$x \$y</code>	True si \$x ou \$y est vrai
!	<code>!\$x</code>	True si \$x n'est pas vrai

53

Instructions conditionnelles

1. If...

- **Syntaxe:**

```
if (condition) {
    code to be executed if condition is true;
}
```

- **Exemple:**

- **Sortie:** « Have a good day! » si l'heure actuelle (\$t) est inférieure à 20 :

```
<?php
$t = date("H");

if ($t < "20") {
    echo "Have a good day!";
}
?>
```

54

Instructions conditionnelles

2. If... else

- **Syntaxe:**

```
if (condition) {
    code to be executed if condition is true;
} else {
    code to be executed if condition is false;
}
```

- **Exemple:**

- **Sortie:** « Have a good day! » si l'heure actuelle (\$t) est inférieure à 20, sinon, « Have a good night! »:

```
<?php
$t = date("H");

if ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

55

Instructions conditionnelles

3. If...elseif

- **Syntaxe:**

```
if (condition) {
    code to be executed if this condition is true;
} elseif (condition) {
    code to be executed if first condition is false and this condition is true;
} else {
    code to be executed if all conditions are false;
}
```

- **Exemple:**

```
<?php
$t = date("H");

if ($t < "10") {
    echo "Have a good morning!";
} elseif ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

56

Instructions conditionnelles

4. Switch

- Selon la valeur d'une variable, un seul bloc sera exécuté,
- Si aucune correspondance entre les blocs et la valeur, le bloc par défaut sera exécuté.

```
<?php
$note = 10;

switch ($note){
    case 0:
        echo 'Vous avez obtenu la note de 0';
        break;
    case 5:
        echo 'Vous avez obtenu la note de 5';
        break;
    case 10:
        echo 'Vous avez obtenu la note de 10';
        break;
    case 15:
        echo 'Vous avez obtenu la note de 15';
        break;
    case 20:
        echo 'Vous avez obtenu la note de 20';
        break;
    default:
        echo 'Je n'ai rien à afficher pour votre note!';
}
?>
```

57

Instructions conditionnelles

4. Switch (Activité)

- **Activité:** Créez une instruction switch qui affichera "Hello" si \$color est "red" et "welcome" si \$color est "green".
- **Solution ?**

58

Les boucles

- **while** - exécute un bloc de code tant que la condition spécifiée est vraie
- **do...while** – exécute un bloc de code une fois, puis répète la boucle tant que la condition spécifiée est vraie
- **for** – exécute un bloc de code un nombre de fois spécifié
- **foreach** – exécute un bloc de code pour chaque élément d'un tableau

59

Les boucles

1. while

- Syntaxe:**

```
while (condition is true) {
    code to be executed;
}
```

- Exemple:**

```
<?php
$x = 1;

while($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
?>
```

Exécution:

```
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
```

60

Les boucles

2. do...while

- Syntaxe:**

```
do {
    code to be executed;
} while (condition is true);
```

- Exemple:**

```
<?php
$x = 1;

do {
    echo "The number is: $x <br>";
    $x+=10;
} while ($x <= 100);
?>
```

Exécution:

```
The number is: 1
The number is: 11
The number is: 21
The number is: 31
The number is: 41
The number is: 51
The number is: 61
The number is: 71
The number is: 81
The number is: 91
```

61

Les boucles

Activité

- Déterminez les diviseurs d'un entier obtenu par tirage aléatoire entre 0 et 1000.
 - Utilisez la fonction `rand(0, 1000)` pour générer le nombre
 - Proposez une solution avec la boucle `while` puis avec la boucle `do-while`.
- Résultat attendu:

les diviseurs de 342 sont:

2, 3, 6, 9, 18, 19, 38, 57, 114, 171,

62

Les boucles

Activité

- **Solution ?**

63

Les boucles

3. for

- **Syntaxe:**

```
for (init counter; test counter; increment counter) {
    code to be executed for each iteration;
}
```

- **Exemple:**

```
<?php
for ($x = 0; $x <= 100; $x+=10) {
    echo "The number is: $x <br>";
}
?>
```

Exécution:

```
The number is: 0
The number is: 10
The number is: 20
The number is: 30
The number is: 40
The number is: 50
The number is: 60
The number is: 70
The number is: 80
The number is: 90
The number is: 100
```

64

Les boucles

4. foreach

- **Syntaxe:**

```
foreach ($array as $value) {
    code to be executed;
}
```

- **Exemple 1:**

```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```

Exécution:

```
red
green
blue
yellow
```


65

Les boucles

4. foreach (suite)

- **Exemple 2:**

```
<?php
$page = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($page as $x => $val) {
    echo "$x = $val<br>";
}
?>
```

Exécution:

Peter = 35

Ben = 37

Joe = 43

66

Exercice 1

- Soit le tableau à deux dimensions suivant:

```
$cars = array (
    array("Volvo",22,18),
    array("BMW",15,13),
    array("Saab",5,2),
    array("Land Rover",17,15)
);
```

- Utilisez la boucle foreach pour afficher \$cars comme suit:

```
Volvo: In stock: 22, sold: 18.
BMW: In stock: 15, sold: 13.
Saab: In stock: 5, sold: 2.
Land Rover: In stock: 17, sold: 15.
```

67

Exercice 1 (suite)

- **Solution ?**

Volvo: In stock: 22, sold: 18.
 BMW: In stock: 15, sold: 13.
 Saab: In stock: 5, sold: 2.
 Land Rover: In stock: 17, sold: 15.

68

Exercice 2

Soit le tableau suivant:

```
<?php
$etudiants=[

    "ali"=>["html"=>[12,15,14], "java"=>[10,18,13], "angular"=>[14,13.5,17]],

    "mohamed"=>["html"=>[13,18,15], "java"=>[9,7,3], "angular"=>[8,7.5,7]],

    "sami"=>["html"=>[17,18,20], "java"=>[14,18,15.5], "angular"=>[13,17.5,18]],

];
```

69

Exercice 2 (suite)

Affichez le bulletin de notes de chaque étudiant sous le format html suivant:

ali

matiere	note tp	note ds	note examen	moyenne
html	12	15	14	13,75
java	10	18	13	13,50
angular	14	13.5	17	15,38
Moyenne generale				14,21
Mention				bien

70

Exercice 2 (suite)

- **Solution ?**

71

Inclusion de fichiers externes

- L'inclusion de fichiers est très utile lorsque vous souhaitez inclure le même PHP, HTML ou texte sur plusieurs pages d'un site Web.
- **Les fonctions utilisées:**
 - `require 'filename'` produira une erreur fatale (E_COMPILE_ERROR) et arrêtera le script si le fichier n'existe pas
 - `include 'filename'` ne produira qu'un avertissement (E_WARNING) et le script continuera si le fichier n'existe pas

72

Inclusion de fichiers externes

- **Exemple:**
 - Supposons que nous ayons un fichier de pied de page standard appelé « footer.php », qui ressemble à ceci :

```
<p> Copyright &copy; 2010-<?=date("Y")?>. isims.rnu.tn </p>
```

- Pour inclure le fichier footer.php dans la page « home.php », utilisez l'instruction include:

```
<html>
<body>

<h1>Welcome to my home page!</h1>
<p>Some text.</p>
<p>Some more text.</p>
<?php include 'footer.php';?>

</body>
</html>
```

Résultat :

Welcome to my home page!

Some text.

Some more text.

Copyright © 2010-2022. isims.rnu.tn

73

Inclusion de fichiers externes

Activité

- Supposons que nous ayons un fichier de menu standard appelé « menu.html », qui ressemble à ceci:

```
<a href="/default.php">Home</a> -
<a href="/html/default.php">HTML Tutorial</a> -
<a href="/css/default.php">CSS Tutorial</a> -
<a href="/js/default.php">JavaScript Tutorial</a> -
<a href="/default.php">PHP Tutorial</a>
```

- Inclure le menu au début de la page « home.php »
 - **Solution ?**

74

Les super-globales

- Sont toujours accessibles, quelle que soit leur portée, depuis n'importe quelle fonction, classe ou fichier.
- Les variables PHP super-globales sont :
 - \$GLOBAUX
 - \$_SERVER
 - \$_REQUEST
 - \$_POST
 - \$_GET
 - \$_FILES
 - \$_ENV
 - \$_COOKIE
 - \$_SESSION

75

Les super-globales

1. \$GLOBALS (rappel)

- PHP stocke toutes les variables globales dans un tableau appelé `$GLOBALS['nom_var']`.
- Ce tableau est également accessible depuis les fonctions et peut être utilisé pour mettre à jour directement les variables globales.

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

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

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

76

Les super-globales

2. \$_SERVER

- Contient des informations sur les en-têtes, les chemins et les emplacements des scripts.
- Pour une liste complète: <https://www.php.net/manual/en/reserved.variables.server.php>

```
<?php
//Renvoie le nom de fichier du script en cours d'exécution
echo $_SERVER['PHP_SELF'];
echo "<br>";
//Renvoie le nom du serveur hôte
echo $_SERVER['SERVER_NAME'];
echo "<br>";
//Renvoie la méthode de requête utilisée pour accéder à la page
echo $_SERVER['REQUEST_METHOD'];
//Renvoie l'en-tête Host de la requête en cours
echo $_SERVER['HTTP_HOST'];
echo "<br>";
//Renvoie l'URL complète de la page en cours
echo $_SERVER['HTTP_REFERER'];
```

Exécution :

```
/VariableServer.php
ex-chap1.test
GET
http://ex-chap1.test/
```

77

Les super-globales

3. \$_REQUEST, \$_POST et \$_GET

- \$_REQUEST, \$_POST et \$_GET sont utilisés pour récupérer des données saisies après avoir soumis un formulaire HTML.

```
<form method="post" action="<?=$_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;
    }
}
?>
```

Exécution :

Name: Envoyer
 Salut ← Après soumission

78

Les super-globales

3. \$_REQUEST, \$_POST et \$_GET

- **Attention !**
 - **\$_REQUEST** peut récupérer les données saisies quelque soit la méthode d'envoi du formulaire (POST ou GET)
 - **\$_POST** peut récupérer uniquement des données d'un formulaire soumis avec une méthode POST
 - **\$_GET** peut récupérer uniquement des données d'un formulaire soumis avec une méthode GET

79

Les superglobales

3. \$_REQUEST, \$_POST et \$_GET

- **Activité:** soit le formulaire suivant

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

- Créez la page welcome_get.php, permettant de récupérer les données saisies dans le formulaire et de les afficher.

- **Exemple d'exécution:**

ex-chap1.test/FormulaireGet.php

Name:

E-mail:

Après soumission

ex-chap1.test/welcome_get.php?name=John&email=john%40gmail.com

Welcome John
Your email address is: john@gmail.com

80

Exercice

1. Créez un formulaire permettant à un membre d'une de ses associations de s'inscrire au repas, de donner ses disponibilités dans un ensemble de dates proposées et de spécifier quel type de plat il préparera.
 - La méthode d'envoi est POST
 - L'action est « inscription.php »

Inscriptions

Prénom :

Mot de passe :

Association :

Disponibilités pour la semaine du 22 juin : ☐ Lundi ☐ Mardi ☐ Mercredi ☐ Jeudi ☐ Vendredi

Contribution : ☐ Entrée ☐ Plat ☐ Dessert

Commentaires :

81

Exercice (suite)

2. Créez le script PHP « inscription.php » permettant d'afficher toutes les informations écrites dans le formulaire et de signaler une erreur si une information est manquante.

82

Les super-globales

4. \$_FILES (téléchargement des fichiers)

- En utilisant le tableau \$_FILES, on peut télécharger des fichiers depuis un ordinateur client vers le serveur distant.
- Les éléments de ce tableau sont comme suit:
 - `$_FILES ["Fichier"] ["nom"]` - le nom du fichier téléchargé
 - `$_FILES ["Fichier"] ["type"]` - le type du fichier téléchargé
 - `$_FILES ["Fichier"] ["size"]` - la taille en octets du fichier téléchargé
 - `$_FILES ["Fichier"] ["tmp_name"]` - le nom de la copie temporaire du fichier stocké sur le serveur
 - `$_FILES ["Fichier"] ["error"]` - le code d'erreur résultant du téléchargement du fichier

83

Les super-globales

4. \$_FILES (téléchargement de fichiers)

- **Démarche à suivre:**

1. Allez à « php.ini » et vérifiez que l'option `file_uploads = On`
2. Créez le formulaire de téléchargement
3. Créer le script PHP du fichier de téléchargement:
 - ×\$_FILES sauvegarde une copie temporaire des fichiers téléchargés dans le dossier temp de PHP sur le serveur.
 - ×Les fichiers temporaires copiés disparaissent lorsque le script se termine.
 - ×Il faut donc stocker le fichier téléchargé et le copier vers un autre emplacement

84

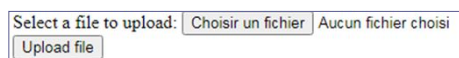
Les super-globales

4. \$_FILES (téléchargement de fichiers)

- **Le formulaire de téléchargement « file_upload.html »**

```
<form action="upload.php" method="post" enctype="multipart/form-data">
  Select a file to upload:
  <input type="file" name="fileToUpload" ><br>
  <input type="submit" value="Upload file" name="upload">
</form>
```

- **Résultat:**



- **Règles à suivre** pour le formulaire HTML ci-dessus :

- Assurez-vous que le formulaire utilise **method="post"**
- Le formulaire a également besoin de l'attribut suivant : **enctype="multipart/form-data"**. Il spécifie le type de contenu à utiliser lors de la soumission du formulaire

85

Les super-globales

4. \$_FILES (téléchargement de fichiers)

- Le script de téléchargement « upload.php »

```
<?php
if ($_FILES["fileToUpload"]["error"] > 0)
{
    echo "Error: " . $_FILES["fileToUpload"]["error"];
}
else
{
    echo "Fichier à télécharger : " . $_FILES["fileToUpload"]["name"] . "<br />";
    echo "Type : " . $_FILES["fileToUpload"]["type"] . "<br />";
    echo "Taille : " . ($_FILES["fileToUpload"]["size"] / 1024) . "Kb<br />";
    echo "Stocké dans : " . $_FILES["fileToUpload"]["tmp_name"];
    //enregistrer le fichier dans le dossier uploads
    if (file_exists("uploads/" . $_FILES["fileToUpload"]["name"]))
    {
        echo "Le fichier " . $_FILES["fileToUpload"]["name"] . " existe déjà à cette emplacement. ";
    }
    else
    {
        move_uploaded_file($_FILES["fileToUpload"]["tmp_name"],
            "uploads/" . $_FILES["fileToUpload"]["name"]);
        echo "Enregistré dans : " . "uploads/" . $_FILES["fileToUpload"]["name"];
    }
}
```

Fichier à télécharger : vecteur2D.py
 Type : application/octet-stream
 Taille : 0.8525390625Kb
 Stocké dans : C:\Users\ACER\AppData\Local\Temp\phpE4E9.tmp
 Enregistré dans : uploads\vecteur2D.py

86

Les super-globales

5. \$_COOKIE

- Qu'est ce qu'un cookie ?

- Un cookie est un petit fichier texte qui permet de stocker une petite quantité de données (près de 4 Ko) sur l'ordinateur de l'utilisateur.
- Il ne pourra par la suite être réutilisé que par le serveur qui l'a déposé.

- Quels sont les usages relatifs aux cookies ?

- Les cookies offrent la possibilité à un site web de:
 - ✗conserver vos préférences,
 - ✗vous garder connecté d'une session sur l'autre
 - ✗ou de vous proposer du contenu personnalisé.

87

Les super-globales

5. \$_COOKIE

- **Création des cookies :**

- Un cookie est créé avec la fonction `setcookie(name, value, expire, path, domain, secure)`
- Seul le paramètre *name* est obligatoire. Tous les autres paramètres sont facultatifs.
- **Exemple:** une cookie nommé « user » avec la valeur « John Doe », et qui expire dans 30 jours

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

88

Les super-globales

5. \$_COOKIE

- **Récupération des cookies:**

- On récupère alors la valeur du cookie "user" (grâce à la variable globale `$_COOKIE`).
- Nous utilisons également la fonction `isset()` pour savoir si le cookie est installé :

```
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

</body>
</html>
```

Cookie 'user' is set!
 Value is: John Doe

89

Les super-globales

5. \$_SESSION

- **Qu'est-ce qu'une session PHP ?**

- Une session est un moyen de stocker des informations (dans des variables) à utiliser sur plusieurs pages.
- Exemple de variables de session: nom d'utilisateur, thème préféré, services accessibles, etc.).
- Par défaut, les variables de session durent jusqu'à ce que l'utilisateur ferme le navigateur.
- Contrairement à un cookie, les informations ne sont pas stockées sur l'ordinateur de l'utilisateur.

90

Les super-globales

5. \$_SESSION

- **Démarrer une session PHP**

- Une session est démarrée avec la fonction `session_start()`.
- Les variables de session sont définies avec la variable globale PHP :
`$_SESSION['nom_var']=valeur`

91

Les super-globales

5. \$_SESSION

- **Démarrer une session**

- **Exemple:** demo1_session.php

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

Attention!

`session_start()` doit être la toute première chose dans votre document, avant toute balise HTML.

92

Les super-globales

5. \$_SESSION

- **Récupérer les valeurs des variables de session PHP**

- Les variables de session ne sont pas transmises individuellement à chaque nouvelle page,
 elles sont récupérées à partir de la session que nous ouvrons au début de chaque page
 (`session_start()`).
- Toutes les valeurs des variables de session sont stockées dans la variable globale `$_SESSION`

93

Les super-globales

5. \$_SESSION

- **Récupérer les valeurs des variables de session**
PHP – exemple: demo2_session.php

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Echo session variables that were set on previous page
echo "Favorite color is " . $_SESSION["favcolor"] . "<br>";
echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
?>

</body>
</html>
```

Pour testez: il faut exécuter:
demo1_session.php
puis demo2_session.php ce qui affiche
en résultat: Favorite color is green.
Favorite animal is cat.

94

Les super-globales

5. \$_SESSION

- **Détruire une session:**

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();
?>

</body>
</html>
```

Exercice

- Créer une page où l'accès n'est autorisé que si on est passé par un formulaire d'authentification.
- **Procédure :**
 - Si le login saisie est « cours2022 » et le mot de passe est « admin », une session est créée et on affecte la valeur "Ok" à la variable de session "Acces".
 - En cliquant sur le lien "continuer" on arrive à la deuxième page du site qui affiche:
 - × « Bonjour » et si on a fourni le code correct, on affiche « l'accès est autorisé ».
 - × Dans le cas contraire, on affiche « l'accès est refusé ».