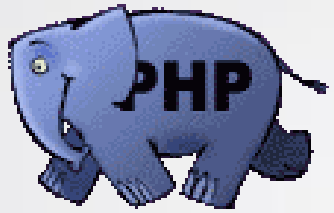
esprit
Se former autrement
HONORIS UNITED UNIVERSITIES



Chapitre 3: PHP

UP Web

AU: 2025/2026

United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
EUR-ACE® Délivrée par la Commission des Titres d'Ingénieur
CONFÉRENCE DES GRANDES ÉCOLES
cdio CONCEIVE DESIGN IMPLEMENT OPERATE

Plan

- Introduction
- Environment de Travail
- Syntaxe
- Tableaux
- Structures Conditionnelles et itératives
- Fonctions

Objectifs

- Comprendre les architectures du web
- Comprendre la syntaxe PHP
- Appréhender les notions de l'orientée objet
- Se connecter à une BD
- Manipuler les données d'une BD via PHP

Prérequis

- Langage HTML

Introduction

Introduction

Navigateur

1. Requête: voir index.php

3. Réponse: Contenu HTML

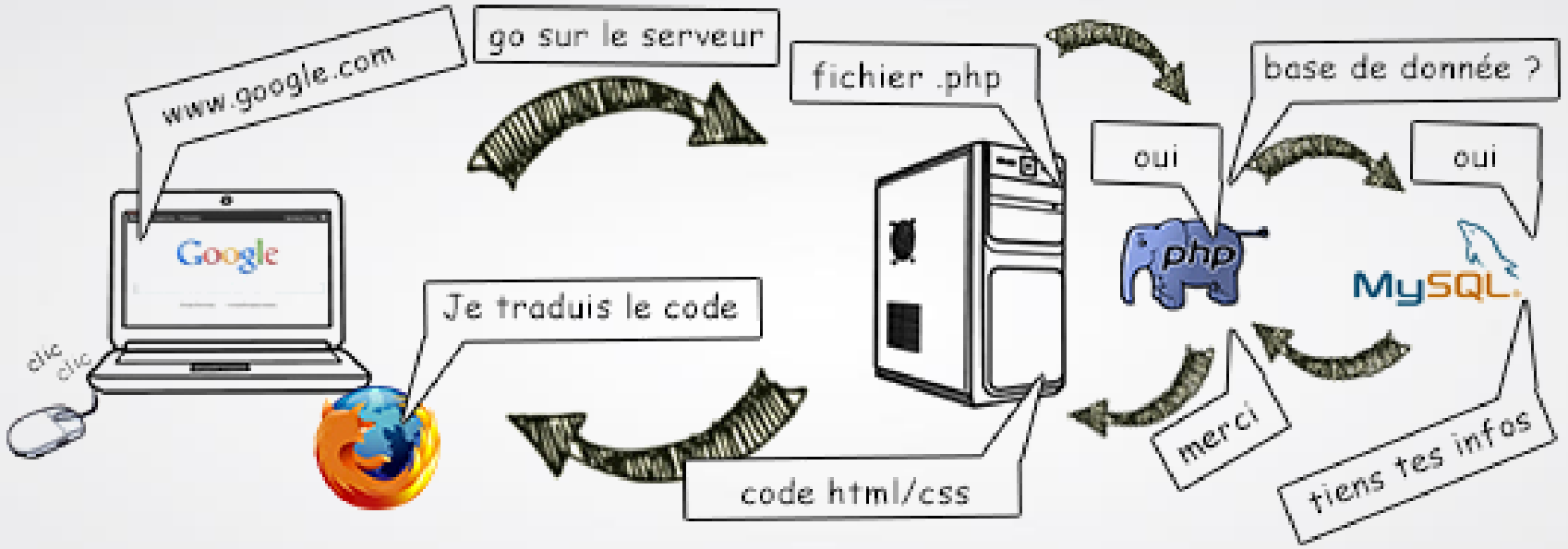
Serveur

4. Affichage:
Interprétation HTML

2. Exécution:
PHP ➔ HTML

Introduction

- **Ce qui se passe réellement**



Introduction

- **PHP**
  - Un langage de programmation qui **s'intègre** dans les pages HTML.
  - Un **langage de scripts** Open Source.
  - Conçu pour le développement d'**applications web dynamiques**.
  - Un langage de script dynamique **précompilé** et **interprété côté serveur**.

Introduction

Branch	Initial Release		Active Support Until		Security Support Until	
7.4	28 Nov 2019	*2 years, 9 months ago*	28 Nov 2021	*9 months ago*	28 Nov 2022	*in 2 months*
8.0	26 Nov 2020	*1 year, 10 months ago*	26 Nov 2022	*in 1 month*	26 Nov 2023	*in 1 year, 1 month*
8.1	25 Nov 2021	*10 months ago*	25 Nov 2023	*in 1 year, 1 month*	25 Nov 2024	*in 2 years, 1 month*

Or, visualised as a calendar:

Key

Active support	A release that is being actively supported. Reported bugs and security issues are fixed and regular point releases are made.
Security fixes only	A release that is supported for critical security issues only. Releases are only made on an as-needed basis.
End of life	A release that is no longer supported. Users of this release should upgrade as soon as possible, as they may be exposed to unpatched security vulnerabilities.

Environnement de Travail

- Pour créer des sites web dynamiques, il est indispensable d'installer un serveur local simulant le travail d'un serveur distant.

- Le serveur local comprend les éléments suivants :
  - Serveur Apache (http://www.apache.org).
  - Interpréteur de code PHP (http://www.php.net).
  - Base de données MySQL (http://www.mysql.com).
  - Utilitaire phpMyAdmin, qui permet de créer et de gérer bases et tables de données MySQL (http://www.phpmyadmin.net).

Environnement de Travail

- Il est plus simple d'installer un paquetage tout prêt:

Syntaxe

Les instructions

```
<?php
// le texte du script php
 ?>
```

Chaque instruction se termine par ;

Les commentaires

- Par ligne:

```
<?php
//commentaire sur une seule ligne
 /*Commentaire sur deux lignes
  */
?>
```

- Ou Par Bloc:

Syntaxe

Les variables

- Le nom d'une variable doit commencer par:
  - Symbole dollar (**$**)
- La déclaration des variables n'est pas obligatoire en début du script.
- Pour afficher le contenu de la variable on utilise le mot clé **echo**

index.php

```php
<?php
    $test = 'Hello World!';
    echo $test;
?>
```

Remarque:

- php est sensible à la casse: maVariable est différente de MaVariable.

Syntaxe

Les variables

- PHP est un langage pauvrement typé, il n'est pas indispensable de déclarer préalablement le type de variable.
- Le type des variables est dynamique, léger et souple.
- Avec PHP nous pouvons utiliser la même variable pour stocker et afficher les différents types de données.
- La variable change de type en fonction du contenu qu'elle reçoit.

Syntaxe

Les variables

- Exemple: typage dynamique

```php
<?php
    $maVariable = 123 ;
    $maVariable = false ;
    $maVariable = 3.1415 ;
    $maVariable = 'Bonjour' ;
?>
```

- Types de données
  - Scalaires
    - Boolean
    - Integer
    - Float
    - String
    - NULL

- Complexes
  - Array
  - Object

Syntaxe

Les constantes

- Une constante est une valeur qui ne pourra plus être modifiée
- Par convention le nom de la constante doit être en majuscule.
- Pour définir une constante, on n'utilise pas le **$**.
- La fonction define retourne TRUE en cas de succès et FALSE en cas de problème.

```php
<?php
    define ('AFFICHE', 'Hello');
    echo AFFICHE;
?>
```

Syntaxe

Les opérateurs

- Les opérateurs servent à comparer, à calculer et à combiner des valeurs.
- Les principaux opérateurs sont:
  - L'opérateur de concaténation: « . »
  - Les opérateurs arithmétiques: «+, -, *, / »
  - Les opérateurs d'affectation.
  - Les opérateurs de comparaison.
  - Les opérateurs logiques.

Syntaxe

Les opérateurs

- Affectation

Exemple	Résultat
$a=1; $b=$a; echo $b;	1
$a=1; echo $a+=1;	2
$a=1; echo $a-=1;	0
$a=1; echo $a*=1;	1
$a=1; echo $a/=1;	1

Syntaxe

Les opérateurs

- Affectation

Exemple	Résultat
$a='Hello'; $b=' ahmed'; echo $a.=$b;	Hello ahmed
$a=1; $a++; echo $a;	2
$a=1; ++$a; echo $a;	2

$a=1;
echo $a++;
echo '
';
echo $a;
echo '
';
echo ++$a;
echo '
';
echo $a;
	1
	2
	3
	3

Syntaxe

Les opérateurs

- Comparaison

Opérateur	Valeur
==	égal
===	identique.
<>	non égal
!=	non égal
!= =	non identique
<	inférieur
>	supérieur
<=	inférieur ou égal
>=	supérieur ou égal

Syntaxe

Les opérateurs

- Comparaison

Opérateur	Valeur
$a and $b	True si $a est true et $b est true
$a or $b	True si $a est true ou $b est true
$a xor $b	True si $a est ou $b est true mais non pas les deux à la fois
! $a	True si $a n'est pas true
$a && $b	True si $a est true et $b est true
$a || $b	True si $a est true ou $b sont true

Les tableaux

Il existe deux types de tableaux:

- **Les tableaux indexés**: Chaque élément du tableau est identifié par son index (0,1,2,..)
- **Les tableaux associatifs**: On associe à chaque élément du tableau une clé dont la valeur est de type chaine de caractères

Les tableaux

Indexés

- Un tableau peut être créé en utilisant array().

- Pour créer un tableau:
  - $arr = array(élément0, élément1, ..., élémentN);
  - $arr = [élément0, élément1, ..., élémentN];
  - $arr = array();

Les tableaux

Indexés

- L'assignation d'une valeur dans un tableau est effectué en spécifiant la clé, entre crochets.
- La clé peut également ne pas être renseignée, sous la forme: [].
  - $arr[clé] = valeur;
  - $arr[] = valeur;
- clé peut être un entier ou une chaîne de caractères (le cas des tableaux associatifs)
- valeur peut être n'importe quel type.

Les tableaux

Indexés

Exercice

- Tester les instructions suivantes:

```php
<?php
    $T=[2,3,6] ;
    print_r($T);
?>
```

```php
<?php
    $T=array();
    $T[]=2;
    $T[]=3;
    $T[]=6;
    print_r($T);
?>
```

```php
<?php
    $T[0]=2;
    $T[1]=3;
    $T[2]=6;
    print_r($T);
?>
```

Les tableaux

Indexés

Exercice

- Tester les instructions suivantes:

```php
<?php
    $T=[2,3,6] ;
    print_r($T);
?>
```

```php
<?php
    $T=array();
    $T[]=2;
    $T[]=3;
    $T[]=6;
    print_r($T);
?>
```

```php
<?php
    $T[0]=2;
    $T[1]=3;
    $T[2]=6;
    print_r($T);
?>
```

Résultat:

```
Array ( [0] => 2 [1] => 3 [2] => 6 )
```

Les tableaux

Associatifs

- Un tableau associatif est un tableau qui va utiliser des clefs textuelles qu'on va associer à chaque valeur.

- Les tableaux associatifs vont s'avérer intéressant lorsqu'on voudra donner du sens à nos clefs, c'est-à-dire créer une association forte entre les clefs et les valeurs d'un tableau

- La création du tableau:
  - en utilisant la structure de langage array()
  - la syntaxe []
  - le construire clef par clef et valeur par valeur.

Les tableaux

Associatifs

- Les clés des tableaux associatifs sont sensibles à la casse.

$T['cle'] est différent de $T['CLE']

- Les chaînes définissants les clés ne doivent pas contenir des espaces.

Les tableaux

Associatifs

Exercice

- Tester les instructions suivantes

```php
<?php
    $personne = array('nom'=>'Ali','Prenom'=>'Salah');
    print_r($personne);
?>
```

```php
<?php
    $personne['nom']='Ali';
    $personne['prenom']='Salah';
    print_r($personne);
?>
```

Les tableaux

Associatifs

Exercice

- Tester les instructions suivantes

```php
<?php
    $personne = array('nom'=>'Ali','Prenom'=>'Salah');
    print_r($personne);
?>
```

```php
<?php
    $personne['nom']='Ali';
    $personne['prenom']='Salah';
    print_r($personne);
?>
```

Résultat

```
Array ( [nom] => Ali [Prenom] => Salah [prenom] => Salah )
```

Les tableaux

Fonctions Prédéfinis

Fonction	Explication
Is_array($T)	Détermine si $T est un tableau
count ($T) \| sizeof ($T)	Retourne le nombre des éléments du tableau
in_array($var, $T)	Teste si la valeur de $var existe dans le tableau $T
sort ($T)	Trie alphanumérique les éléments du tableau et réaffecte les indices du tableau
array_sum($T)	Retourne la somme des valeurs du tableau $T.
array_slice($T, $start [, $nb])	Retourne une portion du tableau commençant à $start et le nombre d'élément $nb si fourni.
array_reverse($T)	Retourne un nouveau tableau qui contient les éléments de $T dans l'ordre inverse.
array_search($value, $T)	Recherche la clé associée à $value.
array_diff($T1, $T2 [,$T3…])	Compare $T1 avec un ou plusieurs tableaux et retourne les valeurs de $T1 qui ne sont pas présentes dans les autres.

Réf: https://www.php.net/manual/fr/ref.array.php

Structures Conditionnelles

If…else

```php
<?php
    if ( condition) {
        // instructions
    }
    else {
        // instructions
    }

?>
```

Structures Conditionnelles

Exercice

- Tester les instructions suivantes:

```php
<?php
    Define('AFFICHE','ch1');
    $ch1='<p>hello</p>';
    $ch2='<p>Bonjour</p>';
    $ch3='<p>Salut</p>';
?>

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
    </head>
    <body>
```

```php
    <?php
        if (AFFICHE=='ch1') {
            echo $ch1;
        }
        else if (AFFICHE=='ch2') {
            echo $ch2;
        }
        else if (AFFICHE=='ch3') {
            echo $ch3;
        }
    ?>
    </body>
</html>
```

Structures Conditionnelles

Switch

- L'instruction switch équivaut à une série d'instructions if .
- Dans une commande switch, une condition n'est évaluée qu'une fois, et le résultat est comparé à chaque case

```php
<?php
    switch (expression) {
        case value_1: // case expression = value_1
            break;
        case value_2: // case expression = value_2
            break;
        case value_3: // case expression = value_3
            break;
        // ...
        default: // default code à exécuter si aucune des autres case n'est vraie
    }
?>
```

Structures Conditionnelles

Exercice

- Tester les instructions suivantes:

```php
<?php
    $colour = "red";
    switch($colour)
    {  case "red":
            $flower = "roses";
            echo $flower;
            break;
        case "blue":
            $flower = "violets";
            echo $flower;
        break;
        default:
            $flower = "unknown";
            echo $flower;
        break;
    }
?>
```

```php
<?php
    $colour = "blue";
    switch($colour):
        case "red":
            $flower = "roses";
            echo $flower;
            break;
        case "blue":
            $flower = "violets";
            echo $flower;
        break;
        default:
            $flower = "unknown";
            echo $flower;
        break;
    endswitch;
?>
```

Structures Itératives

for

```
for ([exp. Initiale]; [Condition]; [incrément]) {
    // instructions
}
```

do…while

```
do {
// instructions
} while (condition);
```

while

```
while (condition) {
// instructions
}
```

Structures Itératives

Exercice

- Tester les instructions suivantes:

```php
for ($i=0; $i < 3; $i++) {
    echo $i. '<br>';
}
```

```php
$i = 0;
do {
    echo $i. '<br>';
    $i++;
} while ($i <= 3);
```

```php
$i = 0;
while ($i <= 3) {
    echo $i. '<br>';
    $i++;
}
```

Structures Itératives

foreach

- Une boucle spécialement dédiée aux tableaux
- Se présente sous 2 syntaxes différentes:

```php
// Affichage des valeurs d'un tableau
foreach ($tab as $value) {
    echo $value, '<br>';
}
```

```php
// Affichage des couples clé / valeur
foreach ($tab as $key => $value) {
    echo $key, ':', $value, '<br>';
}
```

Structures Itératives

Exercice

- Tester les instructions suivantes:

```php
$array = array(
    'premier' => 'N° 1',
    'deuxieme' => 'N° 2',
    'troisieme' => 'N° 3'
);

foreach( $array as $key => $value )
    echo 'Cet élément a pour clé "' . $key . '"
    et pour valeur "' . $value . '"<br />';
```

```php
<?php
    $tab =array(
        'premier'=>'N° 1',
        'deuxieme'=>'N° 2',
        'troisieme'=>'N° 3'
    );
    foreach ($tab  as $value) {
        echo $value . '<br />';
    }
?>
```

Fonction

Syntaxe

```
function nom_fonction ($arg1, $arg2) {
    // instructions
}
```

Exemple

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

Fonction

Fonctions utiles

Fonction	Explication
isset($arg)	permet de savoir si la variable **$arg** existe et initialisé ou pas, **isset** renvoie **TRUE** si la variable est définie et **FALSE** sinon.
empty($arg)	détermine si une variable contient une valeur non nulle. Empty retourne la valeur **FALSE** si la variable **$arg** est affectée ou bien a une valeur différente de 0 et **TRUE** dans les autres cas.
unset($arg)	détruit la variable **$arg**,
var_dump($arg)	affiche le contenu de la variable **$arg** ainsi que son type.

Merci de votre attention