



## ***COURS DE PROGRAMMATION WEB***

Titre du cours	Programmation web
Objectif global	Création d'un site web dynamique avec du PHP
Objectif spécifique	Maitrise les techniques nécessaires à la présentation des documents multimédias -html - CSS maitrisé les techniques a la bases de la création des sites web dynamiques -génération automatique des pages html avec PHP Récupération des donnees utilisateur -créations et interrogation des données Maitrise les techniques permettant d'inter agir avec les clients serveur -Ajax -java script



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

Sous objectif	Contenu clés	Méthodes pédagogiques	Type d'activités	Modalité d'évaluation	Charges du travail
<b>introduction</b>	Définition concept	Actives et interrogative collaboratives	Projet TP TD	Cc sommatives	60 pourcent
<b>Les variables</b>	variables	Actives et interrogative collaboratives	Projet TP TD	Cc sommatives	60 pourcent
<b>Les boucles</b>	Les boucles	Actives et interrogative collaboratives	Projet TP TD	Cc sommatives	60 pourcent
<b>Les conditions</b>		Actives et interrogative collaboratives	Projet TP TD	Cc sommatives	60 pourcent
<b>Les tableaux</b>		Actives et interrogative collaboratives	Projet TP TD	Cc sommatives	60 pourcent
<b>Bases de données</b>			Projet TP TD	Cc sommatives	60 pourcent



Année académique / Academic year : [2020/2021](#)

Centre / Office : [IAI-Cameroun \(Yaoundé\)](#)

Classe / Classroom : [Licence 2](#)

## Evaluation

CC2 +Participation au cours

Cc3+ TP et TD 20points

CC3 +Présence Cours 20points

CC4+ Projet 20points



## CHAPITRE II : PRESENTATION DU LANGAGE

PHP est un langage très souple prenant ses sources dans divers langages comme le C, le Perl, le C++. Il est donc possible d'avoir plusieurs styles de scripts (programmation classique dite procédurale ou programmation objet, ou programmation bordélique). Cette souplesse permet une très grande liberté, un peu comme en Perl. L'inconvénient est qu'on peut très vite obtenir du code illisible(bordélique), même si ça marche très bien. Prenez donc l'habitude de commenter votre code, de l'indenter et de ne placer qu'une instruction par ligne.

### I. Syntaxe de base

#### I.1 Intégration à HTML

Une page php porte l'extension « **.php** ». Une page PHP peut être entièrement programmée en PHP ou mélangée avec du code html. PHP est un langage « **Embedded HTML** », c'est à dire qu'il apparaît à n'importe quel endroit de la page HTML. Pour ça on le place dans des balises particulières : **<?php** et **?>**. On peut aussi utiliser les balises **<script language="php">** et **</script>**. La première forme est préférable pour plus de simplicité et une compatibilité XHTML. On écrit donc une page HTML dans laquelle on intègre du code PHP.

```
<html>
<head>
<title>Titre</title>
</head>
<body>
<?php
echo "HelloWorld!";
```



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

?>

</body>

</html>

Le code HTML généré sera le suivant:

<html>

<head>

<title>Titre</title>

</head>

<body>

**HelloWorld!**

</body>

</html>

L'utilisation de balises pour l'intégration de code dans une page web est très souple et permet de jongler facilement avec du code PHP et du code HTML :

<?php

If(expression) {

?>

<strong>Ceci est vrai.</strong>

<? Php

}else{

?>

<strong>Ceci est faux.</strong>

<?php

}

?>



**NB :** Il existe d'autres balises pour utiliser du PHP, par exemple `<? ?>`, `<% %>`, etc...

## I.2 Séparateur d'instructions

Comme en C une instruction se termine par un point-virgule « ; ». Notez que la balise fermante `?>` implique la fin d'une instruction. Tout langage de programmation contient ce qu'on appelle des **instructions**. On en écrit une par ligne en général, et elles se terminent toutes par un point-virgule. Une instruction commande à l'ordinateur d'effectuer une action précise. Ici, la première instruction que nous allons découvrir permet d'insérer du texte dans la page web. Il s'agit de l'instruction **echo**, la plus simple et la plus basique de toutes les instructions que vous devez connaître

`<?PHP`

`echo "Ceci est un test";`

`?>`

Comme vous le voyez, à l'intérieur de la balise PHP, on écrit l'instruction **echo** suivie du texte à afficher entre guillemets. Les guillemets permettent de délimiter le début et la fin du texte, cela aide l'ordinateur à se repérer. Enfin, l'instruction se termine par un point-virgule comme je vous l'avais annoncé, ce qui signifie Fin de l'instruction.

Notez qu'il existe une instruction identique à **echo** appelée **print**, qui fait la même chose. Cependant, **echo** est plus couramment utilisée.

## I.2 Bloc d'instructions



Un bloc d'instructions se place entre accolades { et }. Un bloc d'instructions peut contenir du code de n'importe quelle longueur et est considéré dans le reste du code comme une instruction unique. Si c'est une expression (qui a une valeur) on peut donc assigner le bloc, faire des calculs, ...

## 4. Commentaires

Un **commentaire** est un texte que vous mettez pour vous dans le code PHP. Ce texte est ignoré, c'est-à-dire qu'il disparaît complètement lors de la génération de la page. Il n'y a que vous qui voyez ce texte. C'est pour vous. Cela permet de vous y retrouver dans votre code PHP, parce que si vous n'y touchez pas pendant des semaines et que vous y revenez, vous risquez d'être un peu perdu. Vous pouvez écrire tout et n'importe quoi, le tout est de s'en servir à bon escient.

Il existe 2 types de commentaires :

- ✓ Les commentaires mono lignes
- ✓ Les commentaires multi lignes

Tout dépend si votre commentaire est court ou long.

### a. *Les commentaires mono lignes*

x

Pour indiquer que vous écrivez un commentaire sur une seule ligne, vous devez taper 2 slashes : //. Tapez ensuite votre commentaire.

### Exemple :

Code : PHP

```
<?php
```

```
echo "J'habite en Chine."; // Cette ligne indique où j'habite
```



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

// La ligne suivante indique mon âge

```
echo "J'ai 92 ans.";
```

```
?>
```

Je vous ai mis deux commentaires à des endroits différents :

- ✓ Le premier est à la fin d'une ligne.
- ✓ Le second est sur toute une ligne

A vous de voir où vous placez vos commentaires : si vous commentez une ligne précise, mieux vaut mettre le commentaire à la fin de cette ligne.

### ***b. Les commentaires multi lignes***

Ce sont les plus pratiques si vous pensez écrire un commentaire sur plusieurs lignes (mais on peut aussi s'en servir pour écrire des commentaires d'une seule ligne). Il faut commencer par écrire `/*` puis refermer par `*/` :

**Code : PHP**

```
<?php
```

```
/* La ligne suivante indique mon âge
```

```
Si vous ne me croyez pas...
```

```
... vous avez raison ; o) */
```

```
echo "J'ai 92 ans.";
```

```
?>
```

Les commentaires s'utilisent comme en C et en C++ avec `/* .. */` et `//`. Notez qu'une balise fermante ferme le commentaire en cours.

## ***II. Les variables***





## 1. Définition

Une variable, c'est une petite information stockée en mémoire temporairement. Elle n'a pas une grande durée de vie. En PHP, la variable (l'information) existe tant que la page est en cours de génération. Dès que la page PHP est générée, toutes les variables sont supprimées de la mémoire car elles ne servent plus à rien. Ce n'est donc pas un fichier qui reste stocké sur le disque dur mais une petite information temporaire présente en mémoire vive. C'est à vous de créer des variables. Vous en créez quand vous en avez besoin pour retenir des informations.

## 2. Les composants d'une variable

Une variable est toujours constituée de deux éléments :

- ✓ **Son nom** : pour pouvoir la reconnaître, vous devez donner un nom à votre variable. Par exemple `age_du_visiteur`.
- ✓ **Sa valeur** : c'est l'information qu'elle contient, qui peut changer. Par exemple : 17.

Ici, je vous ai donné l'exemple d'une variable appelée `age_du_visiteur` qui a pour valeur 17. On peut modifier quand on veut la valeur de cette variable, faire des opérations dessus, etc. Et quand on en a besoin, on l'appelle et elle nous dit gentiment la valeur qu'elle contient.

## 3. Les différents types de variables

Les variables sont capables de stocker différents types d'informations. On parle de **types de données**. Voici les principaux types à connaître :

- **Les chaînes de caractères (string)**: les chaînes de caractères sont le nom informatique qu'on donne au texte. Tout texte est appelé chaîne de caractères. En PHP, ce type de données a un nom : `string`. On peut stocker des textes courts comme très longs au besoin.  
Exemple: "Je suis un texte". Une chaîne de caractères est habituellement écrite entre



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

guillemets ou entre apostrophes (on parle de guillemets simples) : 'Je suis un texte'. Les deux fonctionnent mais il y a une petite différence que l'on va découvrir plus loin.

- **Les nombres entiers (Int)** : ce sont les nombres du type 1, 2, 3, 4, etc. On compte aussi parmi eux les nombres relatifs : -1, -2, -3...Exemple : 42
- **Les nombres décimaux (float)** : ce sont les nombres à virgule, comme 14,738. On peut stocker de nombreux chiffres après la virgule, ce qui devrait convenir pour la plupart des usages que vous en ferez. Attention, les nombres doivent être écrits avec un point au lieu de la virgule (c'est la notation anglaise).Exemple: 14.738
- **Les booléens (bool)**: c'est un type très important qui permet de stocker soit **vrai** soit **faux**. Cela permet de retenir si une information est vraie ou fausse. On les utilise très fréquemment. On écrit true pour vrai, et false pour faux. Exemple : true
- **Rien (NULL)**: aussi bizarre que cela puisse paraître, on a parfois besoin de dire qu'une variable ne contient rien. Rien du tout. On indique donc qu'elle vaut NULL. Ce n'est pas vraiment un type de données, mais plutôt l'absence de type.

En résumé, voici ce qu'il faut retenir des différents types d'informations qu'est capable de stocker PHP dans les variables : Cela devrait vous donner une idée de tout ce qu'est capable de stocker PHP en mémoire.

#### 4. Déclarer une variable

Une variable commence par un dollar « \$ » suivi d'un nom de variable. Les variables ne sont pas typées au moment de leur création. Attention PHP est sensible à la casse : var et Var ne sont pas les mêmes variables ! Voici 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.**

// Déclaration et règles



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

`$var=1;` // \$var est à 1

`$Var=2;` // \$ Var est à 2

`$_toto='Salut';` // Ok

`$3petitscochons=5;` // Invalide : commence par un chiffre

Analysons dans le détail le code qu'on vient de voir :

- D'abord, on écrit le symbole Dollar (\$) : il précède toujours le nom d'une variable. C'est comme un signe de reconnaissance si vous préférez : ça permet de dire à PHP "J'utilise une variable". Vous reconnaîtrez toujours qu'il y a une variable par la présence du symbole Dollar (\$).
- Ensuite, il y a le signe Egal (=) : celui-là c'est logique, c'est pour dire que \$Var/\$\_toto est égal à...
- A la suite, il y a la valeur de la variable, ici 1, 2, salut.
- Enfin, il y a l'incontournable symbole point-virgule (;), qui permet de terminer l'instruction.

## **5. Afficher et concaténer le contenu d'une variable**

### **❖ Afficher le contenu d'une variable**

*Pour afficher la valeur d'une variable ou du texte, il suffit d'utiliser « echo »*

*Exemple*

Code : PHP

`<?php`

`$age_du_visiteur = 17;`



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

`echo $age_du_visiteur;`

`?>`

*Comme vous le voyez, il suffit d'écrire le nom de la variable que vous voulez afficher.*

### ❖ La concaténation

Cela signifie **assemblage**. En fait, écrire 17 tout seul comme on l'a fait n'est pas très parlant. On aimerait écrire du texte autour pour dire : "Le visiteur a 17 ans". La concaténation est justement un moyen d'assembler du texte et des variables. Pour cela, il y a 2 méthodes :

- **Concaténer avec des guillemets doubles**

Avec des guillemets doubles, c'est le plus simple. Vous pouvez écrire le nom de la variable au milieu du texte et elle sera remplacée par sa valeur. Concrètement, essayez ce code :

Code : PHP

```
<?php
```

```
$age_du_visiteur = 17;
```

```
echo "Le visiteur a $age_du_visiteur ans";
```

```
?>
```

ça affiche : Le visiteur a 17 ans. Ça fonctionne bien. En effet, lorsque vous utilisez des guillemets doubles, les variables qui se trouvent à l'intérieur sont analysées et remplacées par leur vraie valeur. Ça a le mérite d'être une solution facile à utiliser mais je vous recommande plutôt la solution qu'on a vu avec des guillemets simples.

- **Concaténer avec des guillemets simples**

Si vous écrivez le code précédent entre guillemets simples, vous allez avoir une drôle de surprise :

Code : PHP



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

```
<?php
```

```
$age_du_visiteur = 17;
```

```
echo'Le visiteur a $age_du_visiteur ans'; // Ne marche pas
```

```
?>
```

Ca affiche : Le visiteur a \$age\_du\_visiteur ans.

Pour que ça marche, il va falloir écrire la variable en dehors des guillemets et séparer les éléments entre eux à l'aide d'un point. Regardez :

Code : PHP

```
<?php
```

```
$age_du_visiteur = 17;
```

```
echo'Le visiteur a ' . $age_du_visiteur . ' ans';
```

```
?>
```

Cette fois, ça affiche bien comme on voulait : Le visiteur a 17 ans. Ça a l'air bien plus compliqué, mais en fait c'est cette méthode qu'utilisent la plupart des programmeurs expérimentés en PHP. En effet, le code est plus lisible, on repère bien la variable alors que tout à l'heure elle était comme "noyée" dans le texte. D'autre part, votre éditeur de texte devrait vous colorier la variable ce qu'il ne faisait pas pour le code précédent.

Il faut noter aussi que cette méthode d'écriture est plus rapide car PHP voit de suite où se trouve la variable et n'a pas besoin de la chercher au milieu du texte.

## **6. Utiliser les types de données**

***Vous vous souvenez des types de données évoqués plus haut : Les string, int, float... Voici un exemple de variable pour chacun de ces types.***

### **a. Le type string (chaîne de caractères)**



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

*Ce type permet de stocker du texte. Pour cela, vous devez entourer votre texte de guillemets doubles "" ou de guillemet simples ' (ce sont des apostrophes). Voici 2 exemples, l'un avec des guillemets simples et l'autre avec des guillemets doubles :*

**Code : PHP**

```
<?php
```

```
$nom_du_visiteur = "Mateo21";
```

```
$nom_du_visiteur = 'Mateo21';
```

```
?>
```

*Attention: si vous voulez insérer un guillemet simple alors que le texte est entouré de*

*guillemets simples, il faut l'échapper en écrivant un antislash devant. De même pour les guillemets doubles. Voici un exemple pour bien comprendre :*

**Code : PHP**

```
<?php
```

```
$variable = "Mon \"nom\" est Mateo21";
```

```
$variable = 'Je m\'appelle Mateo21';
```

```
?>
```

*En effet, si vous oubliez de mettre un antislash, PHP va croire que c'est la fin de la chaîne et il ne comprendra pas le texte qui suivra (vous aurez en fait un message Parse error). Vous pouvez en revanche insérer sans problème des guillemets simples au milieu de guillemets doubles et inversement :*

**Code : PHP**

```
<?php
```

```
$variable = 'Mon "nom" est Mateo21';
```

```
$variable = "Je m'appelle Mateo21";
```



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

?>

*La différence est subtile, faites attention*

**b. Le type `int` (nombre entier)**

*Il suffit tout simplement d'écrire le nombre que vous voulez stocker, sans guillemets.*

**Code : PHP**

<?php

`$age_du_visiteur = 17;`

?>

**c. Le type `float` (nombre décimal)**

*Vous devez écrire votre nombre avec un point au lieu d'une virgule. C'est la notation anglaise.*

**Code : PHP**

<?php

`$poids = 57.3;`

?>

**d. Le type `bool` (booléen)**

*Pour dire si une variable vaut vrai ou faux, vous devez écrire le mot `true` ou `false` sans guillemets autour (ce n'est pas une chaîne de caractères !). Je vous conseille de bien choisir le nom de votre variable pour que l'on comprenne ce que ça signifie. Voyez vous-mêmes :*

**Code : PHP**

<?php

`$je_suis_un_zero = true;`



Année académique / Academic year : 2020/2021

Centre / Office : IAI-Cameroun (Yaoundé)

Classe / Classroom : Licence 2

```
$je_suis_bon_en_php = false;
```

```
?>
```

### e. Une variable vide avec NULL

*Si vous voulez créer une variable qui ne contient rien, vous devez lui passer le mot-clé NULL (vous pouvez aussi l'écrire en minuscules : null).*

Code : PHP

```
<?php
```

```
$pas_de_valeur = NULL;
```

```
?>
```

*Cela sert simplement à indiquer que la variable ne contient rien, tout du moins pour le moment*

Leur type dépend de leur valeur et de leur contexte d'utilisation. Mais on peut forcer (cast) ponctuellement une variable à un type de données, ce qui s'appelle le transtypage. De même comme le type de variable peut changer en fonction de son utilisation ou du contexte, PHP effectue automatiquement un transtypage, ce qui peut parfois fournir des résultats surprenants. On affecte une valeur à une variable avec le signe égal « = » avec ou sans espace avant ou après.

// Déclaration et transtypage

```
$var='2'; // Une chaîne 2
```

```
$var+=1; // $var est maintenant un entier 3
```

```
$var=$var+0.3; // $var est maintenant un réel de type double 3.3
```

```
$var=5 + "3 petits cochons"; // $var est un entier qui vaut 8
```

Par défaut les variables sont assignées par valeur : la valeur assignée à la variable est recopiée dans la variable. PHP peut aussi travailler par référence. Une variable peut donc référencer une autre variable. On dit alors que la variable devient un alias, ou pointe sur une autre variable. On





Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

assigne par référence en utilisant le signe « & » devant la variable assignée

```
$var=2;
```

```
$ref=&$var; // $ref devient une référence de $var
```

```
echo $ref; // affiche 2
```

```
$var=3;
```

```
echo $ref; // affiche 3
```

```
$ref=4;
```

```
echo $var; // affiche 4
```

Attention à la valeur constante NULL insensible à la casse. Affecter une valeur NULL à une variable signifie ne pas lui affecter de valeur.

## 7. Portée des variables

La portée d'une variable dépend du contexte. Une variable déclarée dans un script et hors d'une fonction est globale mais par défaut sa portée est limitée au script courant, ainsi qu'au code éventuellement inclus (include, require) et n'est pas accessible dans les fonctions ou d'autres scripts.

```
$a=1; // globale par défaut
```

```
function foo() {
```

```
echo $a; // c'est une variable locale à la fonction : n'affiche rien
```

```
}
```

Pour accéder à une variable globale dans une fonction, il faut utiliser le mot-clé **global**.

```
$a=1; // globale par défaut
```

```
$b=2; // idem
```



Année académique / Academic year : **2020/2021**

Centre / Office : **IAI-Cameroun (Yaoundé)**

Classe / Classroom : **Licence 2**

```
function foo() {  
  
    global $a,$b; // on récupère les variables globales  
  
    $b=$a+$b;  
}  
echo $b; // affiche 3
```

PHP accepte les variables statiques. Comme en C une variable statique ne perd pas sa valeur quand on sort d'une fonction.

```
function test_static() {  
  
    static $a=0;  
  
    echo $a; // +1 à chaque passage dans la fonction  
  
    $a++;  
}
```

## 8. Variables dynamiques

Une variable dynamique utilise la valeur d'une variable comme nom d'une autre variable. On utilise les variables dynamiques en rajoutant un « \$ » devant le nom de la première variable.

```
$a="var";  
$$a=1; // $$a=1 equivaut en fait à $var=1  
  
echo $a; // affiche var  
  
echo $$a; // affiche 1  
  
echo $var; // affiche 1
```

Attention avec les tableaux ! Pour éviter toute ambiguïté, il est préférable de placer la variable entre accolades.



## 9. Variables prédéfinies

PHP dispose d'un grand nombre de variables prédéfinies. Ces variables sont généralement de types scalaires ou des tableaux. Elles sont souvent de type superglobales, c'est à dire accessible

Depuis n'importe où sans notion de portée. Voici quelques tableaux prédéfinis (voir au point Tableaux pour comprendre leur utilisation).

- ✓ `$_GLOBALS` : tableau des variables globales. La clé est le nom de la variable.
- ✓ `$_SERVER` : variables fournies par le serveur web, par exemple 'SERVER\_NAME'
- ✓ `$_GET` : variables fournies par HTTP par la méthode GET (formulaires)
- ✓ `$_POST` : idem mais pour la méthode POST
- ✓ `$_COOKIE` : les variables fournies par un cookie
- ✓ `$_FILES` : variables sur le téléchargement d'un fichier (upload)
- ✓ `$_ENV` : accès aux variables d'environnement du serveur
- ✓ `$_SESSION` : les variables de session (voir cours sur les sessions)

**Note** : avant la version 4.2.0 de PHP, les variables passées par la méthode GET, POST (formulaires et URL) et ainsi de suite étaient automatiquement créées dans le script de

destination.

**exemple** : `http://www.toto.fr/page.php?action=enr` créait automatiquement la variable `$action` qui prenait la valeur 'enr'. `<input type="hidden" name="cache" value="toto">` créait dans la page cible (action du form) une variable `$cache` de valeur "toto". Ce n'est plus automatique depuis PHP 4.2.0. Il faut passer par les tableaux superglobaux ci-dessus. On peut cependant passer outre en modifiant la valeur register globals dans le `php.ini`.



Année académique / Academic year : [2020/2021](#)

Centre / Office : [IAI-Cameroun \(Yaoundé\)](#)

Classe / Classroom : [Licence 2](#)



## Table des matières

|   |                             |
|---|-----------------------------|
| <b>COURS DE PROGRAMMATION WEB</b>                   | 1                           |
| <b>CHAPITRE II : PRESENTATION DU LANGAGE</b>        | 4                           |
| I. Syntaxe de base                                  | 4                           |
| I.1 Intégration à HTML                              | 4                           |
| I.2 Séparateur d'instructions                       | 6                           |
| I.2 Bloc d'instructions                             | 6                           |
| 4. Commentaires                                     | 7                           |
| a. <i>Les commentaires monolignes</i>               | 7                           |
| b. <i>Les commentaires multilignes</i>              | 8                           |
| 1. <i>Définition</i>                                | 9                           |
| 2. Les composants d'une variable                    | 9                           |
| 3. Les différents types de variables                | 9                           |
| 4. Déclarer une variable                            | 10                          |
| 5. Afficher et concaténer le contenu d'une variable | 11                          |
| 6. Utiliser les types de données                    | 13                          |
| a. Le type string (chaîne de caractères)            | 13                          |
| b. Le type <b>int</b> (nombre entier)               | 15                          |
| c. Le type <b>float</b> (nombre décimal)            | 15                          |
| d. Le type <b>bool</b> (booléen)                    | 15                          |
| e. <i>Une variable vide avec NULL</i>               | 16                          |
| 7. Portée des variables                             | 17                          |
| 8. Variables dynamiques                             | 18                          |
| 9. Variables prédéfinies                            | 19                          |
| <b>PARTIE 1 : LES BASES DU PHP</b>                  | Erreur ! Signet non défini. |