Langage PHP

la liste de codes universels utilisés et qui jusqu'à date n'ont fait aucun défaut:

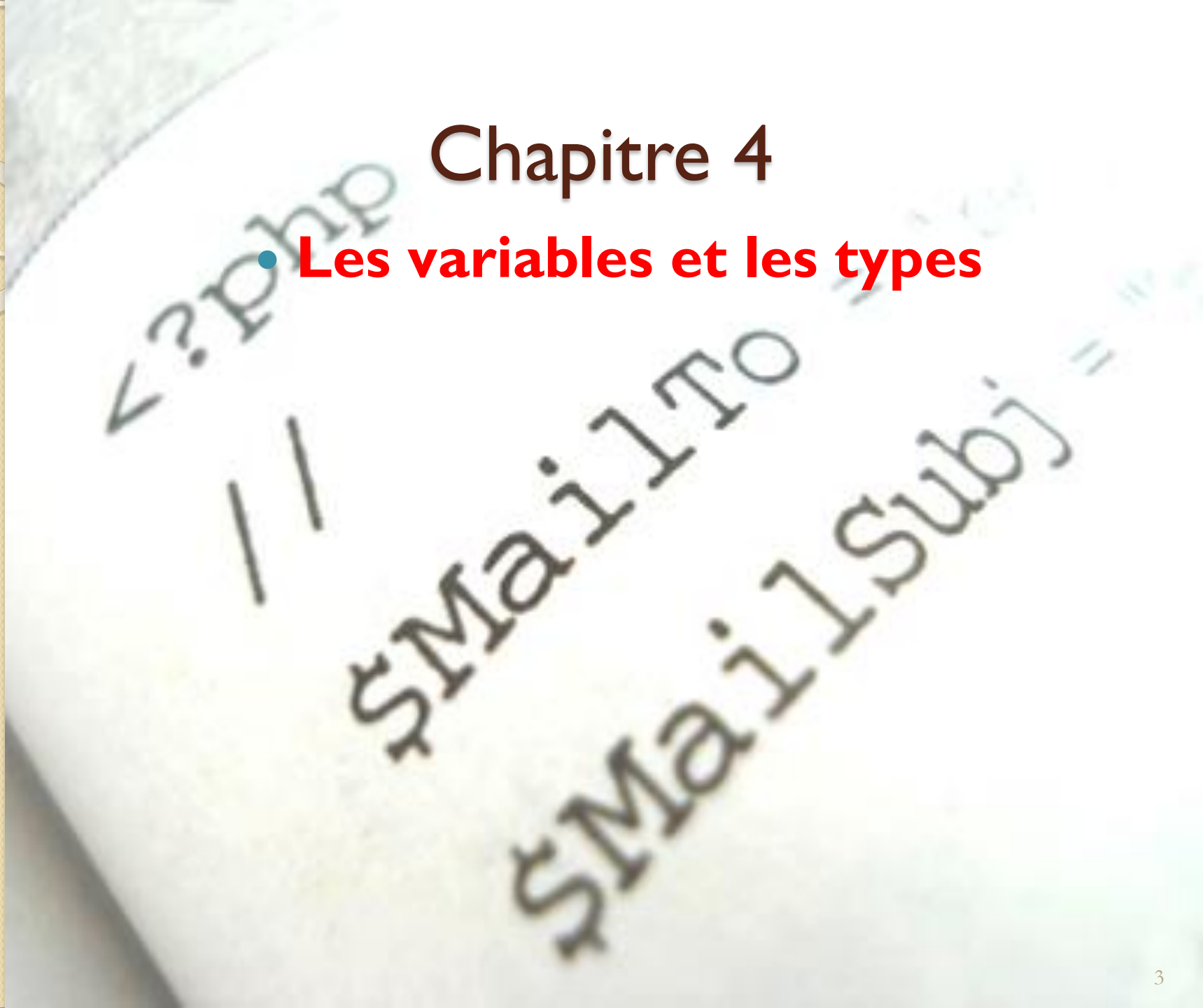
» »
« «
¢ ¢
£ £
© ©
® ®
À À
Á Á
Â Â
Ã Ã
Ä Ä
Ç Ç

È È
É É
Ê Ê
Ë Ë
Ò Ò
Ô Ô
Ù Ù
Ú Ú
Û Û
à à
â â
é é

ê ê
ë ë
î î
ñ ñ
ô ô
ù ù
û û
ù ù
ü ü
ä ä
ç ç
è è

Chapitre 4

- **Les variables et les types**



Introduction

- Les variables sont un élément indispensable dans tout langage de programmation, et en PHP on n'y échappe pas. Ce n'est pas un truc de programmeurs tordus, c'est au contraire quelque chose qui va nous simplifier la vie. Sans les variables, vous n'irez pas bien loin. Ok !!!!

Qu'est-ce qu'une variable ?

- c'est quelque chose qui change tout le temps. En effet, le propre d'une variable c'est de pouvoir **varier.**
- Mais qu'est-ce que c'est concrètement ?
- Une **variable**, c'est une petite information stockée en mémoire **temporairement**. Elle n'a pas une grande durée de vie. En PHP, la variable 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.

Un nom et une valeur

- 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, et qui peut changer. Par exemple : 22.

Les variables (1)

- Principe
  - Commencent par le caractère $
  - N'ont pas besoin d'être déclarées
- Fonctions de vérifications de variables
  - Doubleval(), empty(), gettype(), intval(),
  - is_array(), is_bool(), is_double(), is_float(), is_int(), is_integer, is_long(), is_object(), is_real(), is_numeric(), is_string()
  - Isset(), settype(), strval(), unset()
- Affectation par valeur et par référence
  - Affectation par valeur : $b=$a
  - Affectation par (référence) variable : $c = &$a

Les variables(2)

- Visibilité des variables
  - Variable locale
    - Visible uniquement à l'intérieur d'un contexte d'utilisation
  - Variable globale
    - Visible dans tout le script
    - Utilisation de l'instruction global() dans des contextes locales

```
<?
$var = 100;
function test(){
global $var;
return $var;}
$resultat = test();
if ($resultat) echo $resultat;
else echo " erreur ";
?>
```

Les variables(3)

- Les variables dynamiques
  - Permettent d'affecter un nom différent à une autre variable

```
$nom_variable = 'nom_var';
$$nom_variable = valeur; // équivaut à $nom_var = valeur;
```

  - Les variables tableaux sont également capables de supporter les noms dynamiques

```
$nom_variable = array("val0", "val1", ...,"valN");
${$nom_variable[0]} = valeur; $val0 = valeur;
$nom_variable = "nom_var";
${$nom_variable}[0] = valeur;
$nom_var[0] = valeur;
```

  - Les accolades servent aussi à éviter toute confusion lors du rendu d'une variable dynamique

```
echo "Nom : $nom_variable - Valeur : ${$nom_variable}";
// équivaut à echo "Nom : $nom_variable - Valeur :
  $nom_var";
```

Syntaxe de base : Les variables (4)

- Variables prédéfinies
  - Les variables d'environnement dépendant du client

Variable	Description
$_SERVER["HTTP_HOST"]	Nom d'hôte de la machine du client (associée à l'adresse IP)
$_SERVER["HTTP_REFERER"]	URL de la page qui a appelé le script PHP
$_SERVER["HTTP_ACCEPT_LANGUAGE"]	Langue utilisée par le serveur (par défaut en-us)
$_SERVER["HTTP_ACCEPT"]	Types MIME reconnus par le serveur (séparés par des virgules)
$_SERVER["CONTENT_TYPE"]	Type de données contenu présent dans le corps de la requête. Il s'agit du type MIME des données
$_SERVER["REMOTE_ADDR"]	L'adresse IP du client appelant le script CGI
$_SERVER["PHP_SELF"]	Nom du script PHP

Syntaxe de base : Les variables (5)

- Variables prédéfinies
  - Les variables d'environnement dépendant du serveur

Variable	Description
$_SERVER["SERVER_NAME"]	Le nom du serveur
$_SERVER["HTTP_HOST"]	Nom de domaine du serveur
$_SERVER["SERVER_ADDR"]	Adresse IP du serveur
$_SERVER["SERVER_PROTOCOL"]	Nom et version du protocole utilisé pour envoyer la requête au script PHP
$_SERVER["DATE_GMT"]	Date actuelle au format GMT
$_SERVER["DATE_LOCAL"]	Date actuelle au format local
$_SERVER["$DOCUMENT_ROOT"]	Racine des documents Web sur le serveur

Syntaxe de base : Les variables (6)

- Variables prédéfinies
  - Affichage des variables d'environnement
    - la fonction phpinfo()
      - <? phpinfo(); ?>
      - **echo phpinfo(constante);**

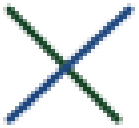
INFO_CONFIGURATION	affiche les informations de configuration.
INFO_CREDITS	affiche les informations sur les auteurs du module PHP
INFO_ENVIRONMENT	affiche les variables d'environnement.
INFO_GENERAL	affiche les informations sur la version de PHP.
INFO_LICENSE	affiche la licence GNU Public
INFO_MODULES	affiche les informations sur les modules associés à PHP
INFO_VARIABLES	affiche les variables PHP prédéfinies.

    - la fonction getenv()
      - <? echo getenv("HTTP_USER_AGENT");?>

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 nombres entiers (int)**
  - **Les nombres décimaux (float)**
  - **Les booléens (bool)**
  - **Rien (NULL)**

Types de données

Type de données	Exemple de valeur
string	`"Du texte"`
int	`42`
float	`14.738`
bool	`true`  `false` 
NULL	

Syntaxe de base : Les types de données

- Principe
  - Pas besoin d'affecter un type à une variable avant de l'utiliser
    - La même variable peut changer de type en cours de script
    - Les variables issues de l'envoi des données d'un formulaire sont du type string
- Les différents types de données
  - Les entiers : le type Integer
  - Les flottants : le type Double
  - Les tableaux : le type array
  - Les chaînes de caractères : le type string
  - Les objets

Syntaxe de base : Les types de données (2)

- Le transtypage
  - La fonction settype() permet de convertir le type auquel appartient une variable

```
<?          $nbre=10;
   Settype($nbre, " double ");
   Echo " la variable $nbre est de type " , gettype($nbre); ?>
```

  - Transtypage explicite : le cast
    - (int), (integer) ; (real), (double), (float); (string); (array); (object)

```
<? $var=" 100 FRF ";
   Echo " pour commencer, le type de la variable est $var, gettype($var);
   $var =(double) $var;
   Echo <br> Après le cast, le type de la variable est $var ", gettype($var);
   Echo "<br> et a la valeur $var "; ?>
```

- Détermination du type de données
  - Gettype(), Is_long(), Is_double(), Is_string(), Is_array(), Is_object(), Is_bool()

Affecter une valeur à une variable

- Premières manipulations de variables
- Utiliser les types de données

Premières manipulations de variables

- Notez qu'on ne peut pas mettre d'espace dans un nom de variable. À la place, utilisez un *underscore* « _ » (c'est le symbole sous le chiffre 8 sur un clavier AZERTY français.
- Pour le nom, évitez aussi les accents, les cédilles et tout autre symbole

- 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 « égal » (=) : celui-là c'est logique, c'est pour dire que $nom est égal à…

- À la suite, il y a la valeur de la variable, ici x.

- Enfin, il y a l'incontournable point-virgule (;) qui permet de terminer l'instruction.

Afficher et concaténer des variables

- Afficher le contenu d'une variable.
  - Exemple :
    - <?php
    - $nom=x;
    - Echo $nom;
    - ?>
- La concaténation
    - <?php
    - $age = 26;
    - **echo** "Le visiteur a ";
    - **echo** $age;
    - **echo** " ans";
    - ?>

- ***Concaténer avec des guillemets doubles.***
  - *Tout simplement* Vous pouvez écrire le nom de la variable au milieu du texte et il sera remplacé par sa valeur.
  - <?php
  - $age = 26;
  - **echo** "Le visiteur a **$age** ans";
  - ?>

Concaténer avec des guillemets simples

- <?php
- $age = 26;
- **echo** 'Le visiteur a $age ans';
- ?>

- <?php
- $age = 26;
- **echo** 'Le visiteur a ' . $age. ' ans';
- ?>

Syntaxe de base : Les chaînes de caractères(1)

Principe

- Peuvent être constituées de n'importe quel caractère alphanumérique et de ponctuation, y compris les caractères spéciaux

```
\t La nouvelle monnaie unique, l' €uro, est enfin là...\n\r
```

- Une chaîne de caractères doit être toujours entourée par des guillemets simples (')ou doubles (")

" Ceci est une chaîne de caractères valide."
'Ceci est une chaîne de caractères valide.'
"Ceci est une chaîne de caractères invalide.'

- Des caractères spéciaux à insérer directement dans le texte, permettent de créer directement certains effets comme des césures de lignes

Car	Code ASCII	Code hex	Description
\car			échappe un caractère spécifique.
" "	32	0x20	un espace simple.
\t	9	0x09	tabulation horizontale
\n	13	0x0D	nouvelle ligne
\r	10	0x0A	retour à chariot
\0	0	0x00	caractère NUL
\v	11	0x0B	tabulation verticale

Syntaxe de base : Les chaînes de caractères(2)

- Quelques fonctions de manipulation

chaîne_result = addCSlashes(chaîne, liste_caractères);
ajoute des slashs dans une chaîne

chaîne_result = addSlashes(chaîne);
ajoute un slash devant tous les caractères spéciaux.

chaîne_result = chop(chaîne);
supprime les espaces blancs en fin de chaîne.

caractère = chr(nombre);
retourne un caractère en mode ASCII

chaîne_result = crypt(chaîne [, chaîne_code])
code une chaîne avec une base de codage.

echo expression_chaîne;
affiche à l'écran une ou plusieurs chaînes de caractères.

$tableau = explode(délimiteur, chaîne);
scinde une chaîne en fragments à l'aide d'un délimiteur et retourne un tableau.

Syntaxe de base : les opérateurs (1)

- Les opérateurs
  - les opérateurs de calcul
  - les opérateurs d'assignation
  - les opérateurs d'incrémentation
  - les opérateurs de comparaison
  - les opérateurs logiques
  - les opérateurs bit-à-dit
  - les opérateurs de rotation de bit

Syntaxe de base : Les opérateurs(2)

- Les opérateurs de calcul

Opérateur	Dénomination	Effet	Exemple	Résultat
+	opérateur d'addition	Ajoute deux valeurs	$x+3	10
-	opérateur de soustraction	Soustrait deux valeurs	$x-3	4
*	opérateur de multiplication	Multiplie deux valeurs	$x*3	21
/	plus: opérateur de division	Divise deux valeurs	$x/3	2.3333333
=	opérateur d'affectation	Affecte une valeur à une variable	$x=3	Met la valeur 3 dans la variable $x

Syntaxe de base : Les opérateurs(3)

- Les opérateurs d'assignation

Opérateur	Effet
+=	addition deux valeurs et stocke le résultat dans la variable (à gauche)
-=	soustrait deux valeurs et stocke le résultat dans la variable
*=	multiplie deux valeurs et stocke le résultat dans la variable
/=	divise deux valeurs et stocke le résultat dans la variable
%=	donne le reste de la division deux valeurs et stocke le résultat dans la variable
|=	Effectue un OU logique entre deux valeurs et stocke le résultat dans la variable
^=	Effectue un OU exclusif entre deux valeurs et stocke le résultat dans la variable
&=	Effectue un Et logique entre deux valeurs et stocke le résultat dans la variable
.=	Concatène deux chaînes et stocke le résultat dans la variable

Syntaxe de base : Les opérateurs(4)

- Les opérateurs d'incrémentation

Opérateur	Dénomination	Effet	Syntaxe	Résultat (avec x valant 7)
++	Incrémentation	Augmente d'une unité la variable	$x++	8
--	Décrémentation	Diminue d'une unité la variable	$x--	6

- Les opérateurs de comparaison

Opérateur	Dénomination	Effet	Exemple	Résultat
==	opérateur d'égalité	Compare deux valeurs et vérifie leur égalité	$x==3	Retourne 1 si $X est égal à 3, sinon 0
<	opérateur d'infériorité stricte	Vérifie qu'une variable est strictement inférieure à une valeur	$x<3	Retourne 1 si $X est inférieur à 3, sinon 0
<=	opérateur d'infériorité	Vérifie qu'une variable est inférieure ou égale à une valeur	$x<=3	Retourne 1 si $X est inférieur à 3, sinon 0
>	opérateur de supériorité stricte	Vérifie qu'une variable est strictement supérieure à une valeur	$x>3	Retourne 1 si $X est supérieur à 3, sinon 0
>=	opérateur de supériorité	Vérifie qu'une variable est supérieure ou égale à une valeur	$x>=3	Retourne 1 si $X est supérieur ou égal à 3, sinon 0
!=	opérateur de différence	Vérifie qu'une variable est différente d'une valeur	$x!=3	Retourne 1 si $X est différent de 3, sinon 0

Syntaxe de base : Les opérateurs(5)

- Les opérateurs logiques

Opérateur	Dénomination	Effet	Syntaxe
|| ou **OR**	OU logique	Vérifie qu'une des conditions est réalisée	((condition1)||(condition2))
&& ou **AND**	ET logique	Vérifie que toutes les conditions sont réalisées	((condition1)&&(condition2))
XOR	OU exclusif	Opposé du OU logique	((condition1)XOR(condition2))
!	NON logique	Inverse l'état d'une variable booléenne (retourne la valeur 1 si la variable vaut 0, 0 si elle vaut 1)	(!condition)

- Les opérateurs bit-à-bit

Opérateur	Dénomination	Effet	Syntaxe	Résultat
&	ET bit-à-bit	Retourne 1 si les deux bits de même poids sont à 1	9 & 12 (1001 & 1100)	8 (1000)
|	OU bit-à-bit	Retourne 1 si l'un ou l'autre des deux bits de même poids est à 1 (ou les deux)	9 | 12 (1001 | 1100)	13 (1101)
^	OU bit-à-bit	Retourne 1 si l'un des deux bits de même poids est à 1 (mais pas les deux)	9 ^ 12 (1001 ^ 1100)	5 (0101)
~	Complément (NON)	Retourne 1 si le bit est à 0 (et inversement)	~9 (~1001)	6 (0110)

Syntaxe de base : Les opérateurs(6)

- Les opérateurs de rotation de bit

Opérateur	Dénomination	Effet	Syntaxe	Résultat
<<	Rotation à gauche	Décale les bits vers la gauche (multiplie par 2 à chaque décalage). Les zéros qui sortent à gauche sont perdus, tandis que des zéros sont insérés à droite	6 << 1 (110 << 1)	12 (1100)
>>	Rotation à droite avec conservation du signe	Décale les bits vers la droite (divise par 2 à chaque décalage). Les zéros qui sortent à droite sont perdus, tandis que le bit non-nul de poids plus fort est recopié à gauche	6 >> 1 (0110 >> 1)	3 (0011)

- Autres opérateurs

Opérateur	Dénomination	Effet	Syntaxe	Résultat
.	Concaténation	Joint deux chaînes bout à bout	"Bonjour"."Au revoir"	"BonjourAu revoir"
$	Référencement de variable	Permet de définir une variable	$MaVariable = 2;	
->	Propriété d'un objet	Permet d'accéder aux données membres d'une classe	$MonObjet->Propriete	

Syntaxe de base : Les opérateurs(7)

- Les priorités

Priorité des opérateurs												
	()	[]										
	--	++	!	~	-							
	*	/	%									
	+	-										
	<	<=	>=	>								
	==	!=										
	&											
	^											
	|											
	&&											
	||											
	?	:										
	=	+=	-=	*=	/=	%=	<<=	>>=	>>>=	&=	^=	|=
	AND											
	XOR											

Résumé

- Une variable est une petite information qui reste stockée en mémoire le temps de la génération de la page PHP. Elle a un nom et une valeur.
- Il existe plusieurs types de variables qui permettent de stocker différents types d'informations : du texte (string), des nombres entiers (int), des nombres décimaux (float), des booléens pour stocker vrai ou faux (bool), etc.
- En PHP, un nom de variable commence par le symbole dollar : $age par exemple.
- La valeur d'une variable peut être affichée avec l'instruction echo.
- Il est possible de faire des calculs mathématiques entre plusieurs variables : addition, soustraction, multiplication…

FIN

- Merci pour votre attention