1 Déclarer des variables

En langage de programmation, une variable correspond à un stockage d'information en mémoire vive (RAM).

Définition générale : une **variable** est un emplacement mémoire repéré par une adresse hexadécimale, un contenu (la donnée à stocker), un type (entier, chaine de caractères, …), et un nom (valable dans un espace de noms). Quand la variable correspond à une instance de classe, on utilise plutôt le mot **objet**.

L'utilisation des variables en PHP est assez souple et peu compliquée.
Exemple : `$age=30;` //suffit à créer une variable nommée « age » contenant 30

En PHP, une variable n'a pas besoin d'un type : le type est automatique et dynamique ; il s'adapte au contenu :

```
$age=20;            // variable nommée « age » contenant 20
var_dump($age);     // Affiche à l'écran :  int(20) c'est un entier
$age="22 ans";      // Redéfinition du contenu de la variable « age »
var_dump($age);     // Affiche à l'écran :  string(6) "22 ans" :  c'est devenu une chaine de car.
```

Le nom d'une variable doit utiliser les caractères du clavier, mais ne jamais commencer par un chiffre :

```
$4saisons= "hiver"      // incorrect
$_4saisons="hiver"      // correct : le premier caractère est _
```

L'adresse de la variable n'est pas directement accessible (comme on le fait avec les pointeurs en langage C). Par contre, on peut utiliser les variable par référence (signe &) :

Exemples :

```
$var1 = 65 ;      // var1 contient 65
$var2 = $var1;    // var2 est une nouvelle variable contenant 65
$var1 = 66;       // var1 contient maintenant 66, var2 contient toujours 65
$var3 = &$var1;   // var3 est une référence (un alias) de la variable var1
$var3 = 77;       // var3 et var1 contiennent 77 alors que var2 contient toujours 65
```

L'appel des variables par référence est souvent utilisé pour les arguments de fonction et dans la programmation objet. Exemple avec une fonction :

Version avec passage par référence & :

```php
<?php
  $a = 5;
  ajouter1( $a );// $a vaut 6 maintenant

  function ajouter1( &$var )
  {
      $var++;
  }

 ?>
```

Version sans la référence & :

```php
<?php
  $a = 5;
  $a = ajouter1( $a );

  function ajouter1( $var )
  {
      return $var++;
  }

 ?>
```

Note : Sans le passage par référence, « var » serait une variable indépendante, existant uniquement dans la fonction. « a » resterait inchangée.

Contenu d'une variable après initialisation : En PHP, contrairement au langage C, le contenu est 0, *false*, ou NULL, suivant le type de variable. Cependant, on veillera à toujours initialiser les variables, par sécurité.
La fonction PHP `isset( $var )` renvoi VRAI (ou 1) si la variable « var » a déjà été initialisée.

2 Afficher une variable

Différentes commandes et variantes sont possibles :
Commande *echo* :

```
echo  $var ;                    // Affiche le contenu de « var ».
echo  "Bonjour" ;              // Affichage d'une chaine de caractères fixe
```

Pour intégrer une variable à une chaine de caractères fixe, il faut faire attention au type de guillemets utilisés :
On peut mettre les variables à l'intérieur des guillemets doubles. Si on met les guillemets simples, on utilise la concaténation avec le « point ».

```
echo  "Texte $var" ;           // Affiche Texte suivi du contenu de « var ».
echo  'Texte ' . $var ;        // Affiche Texte suivi du contenu de « var » (concatenation)
```

Utilitaires :

```
var_dump($var) ;          // Affiche la variable et son contenu
print_r($tableau) ;       // Affiche une variable, même composée (ex : array) de façon lisible.
```

Exercices :

Code php	Affichage
`$var = 3 ;` `echo "Mars est le numéro $var" ;`	
`$var = 3 ;` `echo 'Mars est le numéro $var' ;`	
`$var = 3 ;` `echo 'Mars est le numéro ' . $var ;`	

Exercices :

Code php	Contenu
`$var = 3 ;` `$1erMois = "janvier"`	
`$var = 3 ;` `$var1 = $var + 4 ;`	

`$var = 3 ;`
`$var1 = $var + "4" ;`
`$var1 = $var . "4" ;`	

`$var = "janvier " ;` `$var= $var + " 2023"`	

`$var1 = 5 ;`
`$var2 = &$var1 ;`
`$var1++ ;`
`$var2++ ;`	

`$var11 = 5 ;`
`isset($var11);`
`isset($var22);`
`$var22++ ;`
`isset($var22);`
	Ce isset renvoi :
	Ce isset renvoi :
	
	Ce isset renvoi :

3 $variable : portée

Une variable n'existe pas forcement dans tout le programme. C'est ce qu'on appelle sa « portée ».
Dans un fichier PHP, même si le code est fractionné, noyé dans du HTML, une variable existe dans le code PHP qui la suit. C'est pour cela que c'est une bonne idée de déclarer et initialiser les variables en tout début de la page PHP.

Une variable déclarée en début de page PHP peut être considérée comme globale à la page PHP.
SAUF à l'intérieur d'une fonction (function() {}) :

Une variable déclarée en dehors de la fonction n'est pas connue à l'intérieur des { } d'une fonction. C'est pour cela qu'on utilise le passage d'arguments (par valeur ou par référence).
De la même façon, une variable créée dans une fonction n'est pas connue dans le reste du programme. Elle est locale à la fonction qui l'a créée.

Pour contourner cette limitation, on peut dans une fonction accéder à une variable globale par le tableau associatif $GLOBALS["var"], où « var » est le nom de la variable déclarée en dehors de la fonction. A utiliser avec modération car cela dénote un manque de professionnalisme et d'analyse de la part du programmeur. Il faut privilégier le passage par argument (valeur ou référence) ou le système des « set » et « get » des objets (cas de la POO).

Variables statiques :

Dans une fonction, une variable déclarée *static* sera conservée jusqu'à l'appel suivant de la fonction.

```
<?php
 function compteur() {
   static $cpt=0;

   $cpt++;
   return $cpt;
 }

 compteur();
 echo 'Compteur:'.compteur(); // Affiche : Compteur:2
?>
```

Réinitialisation et durée de vie des variables :

A chaque rechargement de page, les variables sont réinitialisées.
A chaque passage sur une autre page du site, les variables sont perdues.
Si une variable doit exister dans plusieurs pages du site, il faut utiliser les variables superglobales, ou un fichier stocké sur le serveur PHP.

4 Variables Superglobales

Ce sont des variables disponibles sur d'autres pages du site internet, parfois même hors connexion.
Elles sont de type « tableau associatif » (array … toujours avec [ ])

`$_GET["var"]` ou `$_POST["var"]` Utilisées avec les formulaires, transmises d'une page à l'autre par l'URL ou le header.

`$_SERVER["…"]` Variables générées par le système. Des informations sur l'environnement html/php.
Ex : `$_SERVER["REMOTE_ADDR"]` : adresse ip du client. Utile pour tracer un visiteur.

`$_COOKIE["var"]` Variable stockée localement par le navigateur jusqu'à expiration d'un délai.

`$_SESSION["var"]` Variable stockée sur le serveur PHP, valable dans toutes les pages du site tant que le navigateur n'est pas fermé.

5 Les Cookies

Grâce à la fonction *setcookie()*, le serveur PHP peut stocker sur l'ordinateur du client un fichier contenant une information ou un tableau (*array*) d'informations. Cette opération doit être effectuée avant la balise <html> dans le cas d'une page mixte, car l'ordre de création du cookie est envoyé dans l'en tête http (header).

CONSEQUENCE : Sur une même page PhP, on ne peut pas créer un cookie et l'utiliser immédiatement. Il n'existera qu'au prochain chargement de la page. Par contre il est possible sur une même page d'utiliser du *Javascript* à la place du PhP pour tester la présence du cookie (http://www.w3schools.com/js/js_cookies.asp) puisque le *javascript* est exécuté *après* le chargement du *header*.

Exemple d'utilisation des cookies en PHP :

Un cookie simple valable 1 heure : `setcookie("nom","Dupont", time()+3600) ;`
Le fichier cookie est une variable « nom » contenant la valeur « Dupont »
Consulter les documentations en ligne pour voir toutes les options possibles.

Pour utiliser la valeur contenue dans le cookie, une autre page du même site peut faire :

```
if ( isset($_COOKIE["nom"]) )
            echo $_COOKIE["nom"];
else
            echo "pas de cookie NOM";
```

Pour modifier un cookie (ou le supprimer), on utilise également la fonction *setcookie()*.

Les cookies sont très utilisés pour surveiller l'activité des internautes. La plupart des sites en déposent à votre insu et stockent vos centres d'intérêts, afin de mieux les réutiliser lors de votre prochaine visite. On peut demander au navigateur de les supprimer ou de les bloquer. Cependant, certains sites refusent les navigateurs qui n'autorisent pas la création de cookies.

Maintenant les cookies sont stockés sur la machine client dans un fichier base de données SQLite. Auparavant les cookies étaient de simples fichiers stockés localement et visibles par d'autres utilisateurs de la machine. Dans un cas comme dans l'autre, ils nuisent à la confidentialité et au respect de la vie privée.

6 Les sessions

Les variables de session sont stockées sur le serveur PHP uniquement. Le programmeur dispose là d'un moyen sécurisé pour stocker temporairement des données qui seront utilisées pendant la navigation sur l'ensemble du site. Exemple : le nom d'utilisateur saisi par le client. Si le navigateur se ferme, le serveur considère que la session est terminée et les données de session sont effacées ce qui offre un bon niveau de sécurité et de confidentialité pour le client.

Pour activer les variables de session, on utilise la fonction *session_start()* avant la balise <html> car l'identifiant de session est transmis dans l'en-tête http au navigateur.
Les variables de session sont stockées dans le tableau associatif **$_SESSION[ ]**
Pour forcer la fin d'une session et détruire les données, on utilise la fonction *session_destroy()*.

Créer une variable de session :

```php
<?php
  session_start();

  $_SESSION["nom"]="Dupont";
?>
```

Afficher la variable de session :

```php
<?php
  session_start();

  echo "<html> <body>";
  echo "Variable lue:";
  if (isset($_SESSION["nom"]))
       echo $_SESSION["nom"];
  else
       echo "Pas de variable";

  echo "</body></html>";
?>
```

Détruire la session :

```php
<?php
  session_start();
  $_SESSION[]="" ;
  session_destroy();
?>
```

Note : Le SID (Session Identifier) est stocké en local par le navigateur dans un cookie et joint à l'en-tête http à chaque demande de page sur le serveur. Si le navigateur n'accepte pas les cookies, le programmeur du site doit manuellement intégrer le SID à l'en-tête http (voir les options de la fonction *header*). Cela reste un point faible pour les attaques. L'utilisation des cookies de session rend le site moins vulnérable aux attaques.