PHP 5

Table des matières

PHP: Hypertext Preprocessor est un langage de programmation libre principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP.

1. Les sites statiques et dynamiques

Il existe deux types de sites web :

- Les sites statiques : leur contenu ne peut pas être mis à jour automatiquement. Ce sont des sites réalisés uniquement à l'aide des langages HTML et CSS.
- Les sites dynamiques : Le contenu de ces sites web est dit « dynamique » parce qu'il peut changer. Plus complexes, ils utilisent d'autres langages en plus de HTML et CSS, tels que PHP et MySQL.

Lorsque le site est dynamique :

1. Le client demande au serveur à voir une page web ;
2. le serveur prépare la page spécialement pour le client ;
3. le serveur lui envoie la page qu'il vient de générer.

2. Les langages du web

Quel que soit le site web que l'on souhaite créer, HTML et CSS sont indispensables mais ils ne suffisent pas pour réaliser des sites dynamiques. Il faut les compléter avec d'autres langages tel que PHP qui permet de génèrer une page web. Le PHP est un langage de programmation, ce qui n'est pas le cas du HTML (on parle plutôt de langage de description, car il permet de décrire une page web).

D'autres langages existent également :

- ASP .NET : conçu par Microsoft, il exploite le framework (c'est-à-dire un ensemble de bibliothèques qui fournissent des services pour les développeurs) .NET bien connu des développeurs C#.
- Ruby on Rails : ce framework s'utilise avec le langage Ruby et permet de réaliser des sites dynamiques rapidement.
- Django : il est similaire à Ruby on Rails, mais il s'utilise en langage Python.
- Java et les JSP (Java Server Pages) : plus couramment appelé « JEE ».

3. Premiers pas en PHP

3.1. Les balises PHP

Le code PHP vient s'insérer au milieu du code HTML. Pour utiliser du PHP, on doit utiliser la balise qui commence par <?php et se termine par ?> ; c'est à l'intérieur que l'on mettra du code PHP.

Exemple : <?php /* Le code PHP se met ici */ ?>

3.2. Afficher du texte

PHP dispose de deux instructions :

- echo : <?php echo "Ceci est du texte"; ?>

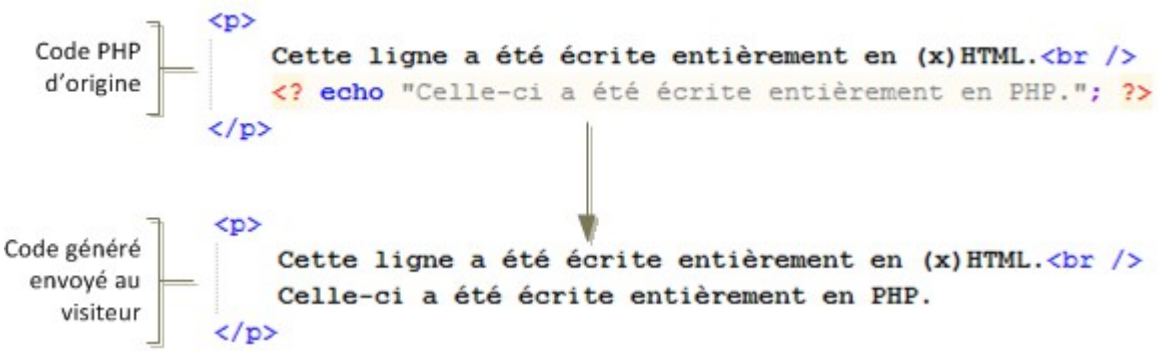
  ou

- print : <?php print("Ceci est du texte"); ?>

Les guillemets permettent de délimiter le début et la fin du texte. Pour afficher un guillemet dans le texte, la solution consiste à le faire précéder d'un antislash \ :

Une instruction se termine par un point-virgule, ce qui signifie Fin de l'instruction.

Le code PHP est exécuté côté serveur. Une fois toutes les instructions PHP exécutées, la page qui sort est une page qui ne contient que du HTML ! C'est cette page de « résultat » qui est envoyée au visiteur, car celui-ci ne sait lire que le HTML.



3.3. Les commentaires

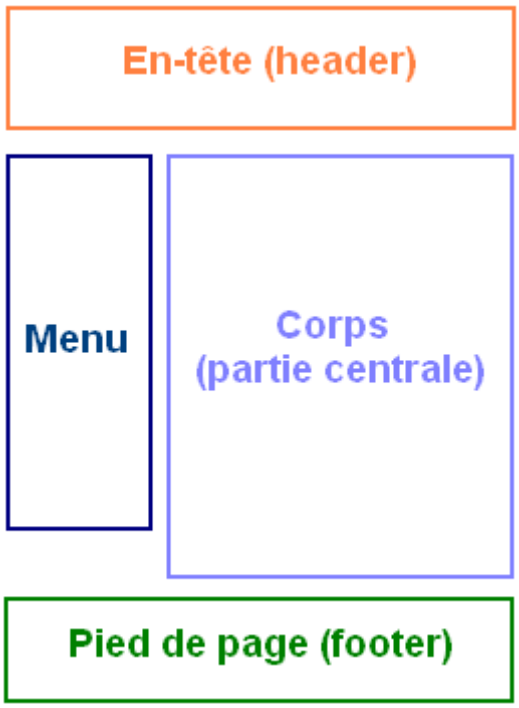
Un commentaire est un texte que vous mettez pour vous dans le code PHP. Ce texte est ignoré, c'est-à-dire qu'il disparaît complètement lors de la génération de la page.

Il existe deux types de commentaires :

- les commentaires monolignes : précédé par //
- les commentaires multilignes : encadré par /* et */

4. Inclure des portions de page

La plupart des sites web sont généralement découpés selon le schéma suivant.



Jusqu'ici, il fallait copier sur chaque page à l'identique :

- l'en-tête ;
- le menu ;
- le pied de page.

Cela qui donnait du code lourd et répétitif sur toutes les pages car d'une page à l'autre ce site contiendra à chaque fois le même code pour l'en-tête, le menu et le pied de page. Seul le contenu du corps change en temps normal.

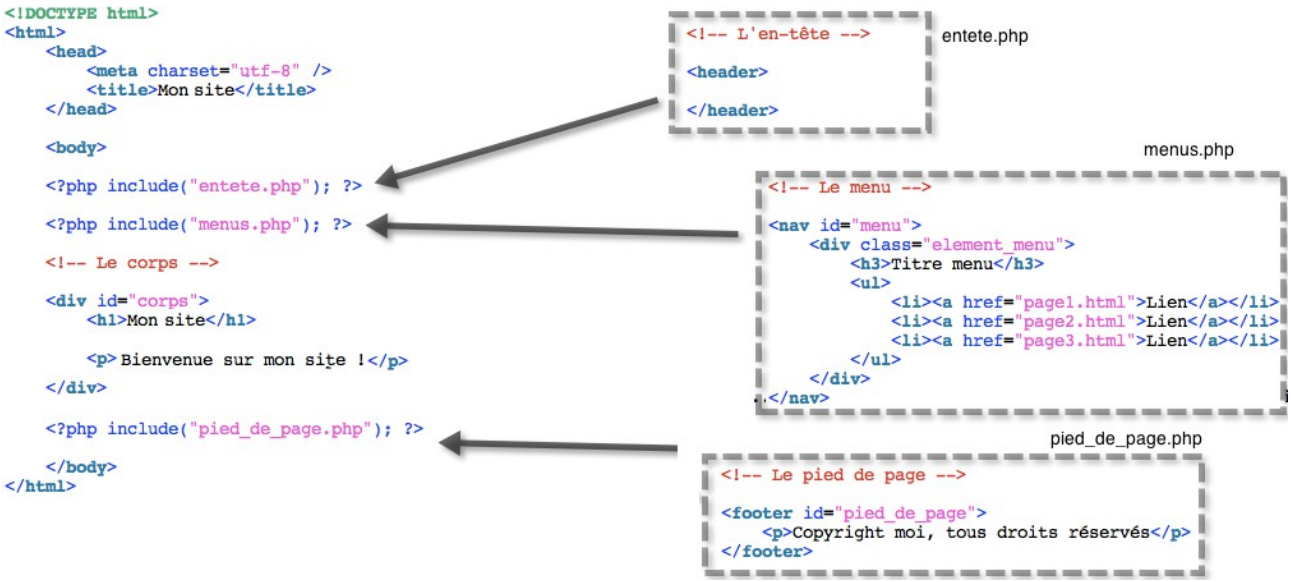
PHP permet d'insérer d'autres pages à l'intérieur d'une page grâce à l'instruction include.

Exemple :

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="utf-8" />
		<title>Mon site</title>
	</head>

	<body>
		<?php include("entete.php"); ?>

		<?php include("menus.php"); ?>
```

```
        <!-- Le corps -->
        <div id="corps">
            <h1>Mon site</h1>

            <p>Bienvenue sur mon site !</p>
        </div>

        <?php include("pied_de_page.php"); ?>
    </body>
</html>
```

Le schéma suivante montre comment les pages sont incluses.



Ce code suppose que la page index.php et celles qui sont incluses sont dans le même dossier. Si le menu était dans un sous-dossier appelé inc, il aurait fallu écrire : <?php include("inc/menus.php"); ?>

Cette technique permet de centraliser le code des menus alors qu'auparavant il fallait le copier dans chaque page statiques en HTML et CSS !

5. Les variables

5.1. Définition d'une variable

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.

Une variable est toujours constituée de deux éléments :

- son nom : pour pouvoir la reconnaître, vous devez donner un nom à votre variable.
- sa valeur : c'est l'information qu'elle contient, et qui peut changer.

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.

  Une chaîne de caractères est habituellement écrite entre guillemets ou entre apostrophes.
- Les nombres entiers (int) : on compte aussi parmi eux les entiers relatifs : -1, -2, -3…
- Les nombres décimaux (float) : ce sont les nombres à virgule. Les nombres doivent être écrits avec un point au lieu de la virgule (c'est la notation anglaise).
- Les booléens (bool) : c'est un type qui ne permet de stocker que deux valeurs, soit vrai (true) soit faux (false).
- Rien (NULL) : ce n'est pas vraiment un type de données, mais plutôt l'absence de type.

5.2. Affecter une valeur à une variable

Le symbole « dollar » ($) : il précède toujours le nom d'une variable .

On ne peut pas mettre d'espace dans un nom de variable, à la place utilisez un underscore « _ » (évitez aussi les accents, les cédilles et tout autre symbole).

```
$age_du_visiteur = 17;          // La variable est créée et vaut 17
$garcon = true;                 // variable booléenne à vrai (le visiteur est un garçon)
$pas_de_valeur = NULL;
$nom_du_visiteur = 'Dom';
$nom_du_visiteur = "Dom";
```

Attention : pour insérer un guillemet simple alors que le texte est entouré de guillemets simples, il faut l'échapper comme on l'a vu précédemment en insérant un antislash devant. Il en va de même pour les guillemets doubles.

```
$nom_du_visiteur = 'Je m\'appelle Dom';
$nom_du_visiteur = "Je m'appelle \"Dom\"";
```

5.3. Afficher et concaténer des variables

Les commandes echo ou print permettent d'afficher la valeur d'une variable.

```
print("Le visiteur a $age_du_visiteur ans");
print("Le visiteur a ".$age_du_visiteur." ans");
echo "Le visiteur a $age_du_visiteur ans";
echo 'Le visiteur a ' . $age_du_visiteur . ' ans';
```

- En utilisant des guillemets doubles, les variables qui se trouvent à l'intérieur sont analysées et remplacées par leur vraie valeur.

- L'opérateur . permet de concaténer (assembler) du texte.

5.4. Calculs simples

Les signes à connaître pour faire les quatre opérations de base sont représentés par le tableau suivant.

Symbole	Signification
+	Addition
-	Soustraction
*	Multiplication
/	Division
%	Modulo
!	Non logique

```
$nombre = 2 + 4;        // $nombre prend la valeur 6
$nombre = 3 * 5 + 1;    // $nombre prend la valeur 16
$nombre = 10 % 3;       // $nombre prend la valeur 1 car il reste 1
```

Calculs avec plusieurs variables :

```
<?php
$nombre = 10;
$resultat = ($nombre + 5) * $nombre;          // $resultat prend la valeur 150
?>
```

6. Les conditions

Une condition peut être écrite en PHP sous différentes formes. On parle de structures conditionnelles.

6.1. if… else

Les symboles suivant permettent de faire des comparaisons simples ou multiples pour les conditions.

Symbole	Signification
==	Est égal à
>	Est supérieur à
<	Est inférieur à
>=	Est supérieur ou égal à
<=	Est inférieur ou égal à
!=	Est différent de

Symbole	Signification
AND	ET logique
&&	ET logique
OR	OU logique
||	OU logique

Pour introduire une condition :

1. on utilise le mot if, qui en anglais signifie « si ».
2. on ajoute à la suite entre parenthèses la condition en elle-même.
3. on ouvre des accolades à l'intérieur desquelles on placera les instructions à exécuter si la condition est **vraie**.
4. **Facultativement** : on utilise le mot else, qui en anglais signifie « sinon ».
5. puis on ouvre des accolades à l'intérieur desquelles on placera les instructions à exécuter si la condition est **fausse**.

```
<?php
$age = 8;                       // initialisation de la variable

if ( $age < 12 )                // test
{
        echo "Salut gamin !";   // action si condition vraie
}
else    // SINON
{
        // action si condition fausse
}
?>
```

NB : au départ, une variable ne contient rien. Pour vérifier si la variable est vide, il suffit d'écrire le test : if ($variable == NULL)…

Le booléens sont des variables qui valent soit true (vrai) soit false (faux). Le test conditionnel est très simple :

```
<?php
if ( $garcon == true )
{
        echo "Bonjour Monsieur";
}
?>
```

ou encore plus simplement :

```
<?php
if ( $garcon )
{
        echo "Bonjour Monsieur";
}
?>
```

Dans le cas des conditions multiples, on utilise les opérateurs logiques.

```
<?php
if ( $age < 12 AND $garcon == true )        // si j'ai moins de 12 ans et que je suis un garçon
{
```

```
        echo "Bonjour Jeune homme";
}
else
 {
        if ( $age < 12 AND $garcon == false )        // si j'ai moins de 12 ans et que je suis une fille
        {
                echo "Bonjour Mademoiselle";
        }
}
?>
```

ou encore plus simplement :

```
<?php
if ( $age < 12 AND $garcon == true )         // si j'ai moins de 12 ans et que je suis un garçon
{
        echo "Bonjour Jeune homme";
}
elseif ( $age < 12 AND $garcon == false )   // si j'ai moins de 12 ans et que je suis une fille
{
        echo "Bonjour Mademoiselle";
}
?>
```

6.2. switch... case

Les structures à base de if… elseif… else suffisent pour traiter n'importe quelle condition.

Cependant, pour une imbrication de if... else trop importante, il existe une autre structure plus souple : switch... case.

```
<?php
$jour = "dimanche";

switch ( $jour )          // on indique sur quelle variable on travaille
{
        case "lundi" :            // dans le cas où c'est le début de la semaine
        case "mardi" :            // on peut tester deux valeurs à la suite
                echo "courage !!!";
        break;                    // ne pas oublier le mot-clef break sinon PHP continue

        case "mercredi" :         // dans le cas où c'est le début de la semaine
                echo "c'est le jour des enfants";
        break;

        case "jeudi" :            // dans le cas où c'est la fin de la semaine
        case "vendredi" :
                echo "bientôt le we !";
        break;
```

```
        default:                    // il faut traiter les autres jours (cas par défaut)
                echo "vive le week end !";
        break;
}
?>
```

Le mot-clé default est le traitement par défaut quelle que soit la valeur de la variable.

PHP traite les instructions des case à la suite. Pour interrompre le traitement, il faut utiliser break.

6.3. Les ternaires : des conditions condensées

Un ternaire est une condition condensée qui fait deux choses sur une seule ligne :

- on teste la valeur d'une variable dans une condition ;
- on affecte une valeur à une variable selon que la condition est vraie ou non.

C'est l'écriture condensée de if… else.

```
<?php
$age = 24;
if ( $age >= 18 )
{
        $majeur = true;
}
else
{
        $majeur = false;
}
?>
```

ou encore plus simplement :

```
<?php
$age = 24;
$majeur = ( $age >= 18 ) ? true : false ;
?>
```

7. Les boucles

7.1. Tant que : while

Une boucle permet de répéter des instructions plusieurs fois.

- les instructions sont d'abord exécutées dans l'ordre, de haut en bas
- à la fin des instructions, on retourne à la première
- et ainsi de suite...

Tant que la condition est remplie, les instructions sont réexécutées. Dès que la condition n'est plus remplie, on sort de la boucle.

```
<?php
$nombre_de_lignes = 1;

while ( $nombre_de_lignes <= 100 )          // on boucle tant que on n'est pas arrivé à 100 lignes
{
        echo "ligne n° $nombre_de_lignes.<br />"; // affiche le n° de ligne
        $nombre_de_lignes++;                      // $nombre_de_lignes = $nombre_de_lignes + 1
}
?>
```

Il faut TOUJOURS s'assurer que la condition sera fausse au moins une fois. Si elle ne l'est jamais, alors la boucle s'exécutera à l'infini !

PHP refuse normalement de travailler plus d'une quinzaine de secondes. Il s'arrêtera tout seul s'il voit que son travail dure trop longtemps et affichera un message d'erreur.

7.2. Faire tant que : do... while

Les boucles do-while ressemblent beaucoup aux boucles while, mais l'expression est testée à la fin de chaque itération plutôt qu'au début. La principale différence par rapport à la boucle while est que la première itération de la boucle do-while est toujours exécutée au moins une fois (l'expression n'est testée qu'à la fin de l'itération).

```
<?php
$nombre_de_lignes = 1;

do
{
        echo "ligne n° $nombre_de_lignes.<br />"; // affiche le n° de ligne
        $nombre_de_lignes++;                      // $nombre_de_lignes = $nombre_de_lignes + 1
} while ( $nombre_de_lignes <= 100 );       // on boucle tant que on n'est pas arrivé à 100 lignes
?>
```

7.3. Boucle : for

La boucle for est un autre type de boucle, dans une forme un peu plus condensée et plus commode à écrire, ce qui fait que for est assez fréquemment utilisé.

for (initialisation ; condition ; incrémentation)

1. initialisation. C'est la valeur que l'on donne au départ à la variable.
2. condition. Comme pour le while, tant que la condition est remplie, la boucle est réexécutée. Dès que la condition ne l'est plus, on en sort.
3. incrémentation, qui vous met à jour la variable à chaque tour de boucle.

```
<?php
for ($nombre_de_lignes = 1 ; $nombre_de_lignes <= 100 ; $nombre_de_lignes++)
{
        echo "ligne n° $nombre_de_lignes.<br />"; // affiche le n° de ligne
}
```

?>

Cette boucle est utilisée lorsque l'on connait à l'avance le nombre d'instructions à répéter.

8. Les fonctions

Une fonction est une série d'instructions qui effectue des actions qui sont répétées plusieurs fois dans le code sans avoir à les réécrire à chaque fois.

Lorsqu'on appelle une fonction, il y a trois étapes.

1. L'entrée: on donne des informations à la fonction en lui passant des paramètres avec lesquelles travailler).
2. Le traitement : grâce aux informations qu'elle a reçues en entrée.
3. La sortie : une fois qu'elle a fini son traitement, la fonction renvoie un résultat.

La syntaxe pour coder un fonction est donc :

```
function nomFonction(parametres)
{
   // Insérez vos instructions ici
}
```

- function est le mot clef qui déclare la fonction
- nomFonction : c'est le nom de la fonction (pas d'accents, pas d'espaces, etc.)
- parametres (correspond à l'entrée) : valeurs avec lesquelles la fonction va travailler. Il peut ne pas y avoir de paramètres.

```
<?php
function cube($la_valeur)     // la fonction est déclarée par le mot-clef function
{                             // début de la fonction
// fonction :    calcule le cube d'une valeur passée en paramètre
// $la_valeur : valeur à calculer
// retour :      résultat du paramètre élevé au cube

        $total = $la_valeur * $la_valeur * $la_valeur ;
        return $total ;       // si la fonction retourne un résultat il faut utiliser le mot-clef return
}                             // fin de la fonction

$valeur = cube(2) ;                          // appel de la fonction
echo "la valeur de 2^3 est $valeur<br/>" ;   // affiche la valeur de 2^3
```

```
echo "la valeur de 3^3 est ".cube(3)."<br/>" ;          // et plus simplement pour 3³
?>
```

Exemple : on veut afficher le cube des nombres de 1 à 10.

```
<?php
function cube($la_valeur)
{
// fonction :    affiche le cube d'une valeur passée en paramètre
// $la_valeur : valeur à calculer
// retour :       rien

        $total = $la_valeur * $la_valeur * $la_valeur ;
        echo "la valeur de $la_valeur^3 est $valeur<br/>" ;
}

// programme principal
for ($i = 1 ; $i <= 10 ; $i++)
{
        cube($i) ;                                          // appel de la fonction
}
?>
```

Une fonction peut recevoir plusieurs paramètres séparés par des virgules.

```
<?php
function puissance($la_valeur, $exposant)
{
// fonction :    calcule la valeur d'un nombre élevé à une puissance
// $la_valeur : valeur à calculer
// $exposant :  puissance
// retour :       le nombre élevé à la puissance

        $total = 1 ;
        for ($i = 1 ; $i <= $exposant ; $i++)
                $total = $total * $la_valeur;

        return $total;
}
?>
```

8.1. Les fonctions PHP

PHP propose des centaines et des centaines de fonctions prédéfinies. La documentation PHP les répertorie toutes, classées par catégories.

Voici un petit aperçu des fonctions qui existent :

- recherche et de remplacement des mots dans une variable.

- envoie un fichier sur un serveur.
- création des images miniatures (aussi appelées thumbnails).
- chiffrement des mots de passe.
- ….

Ainsi la fonction puissance que l'on a codée précédemment existe ! C'est la fonction pow.

Il faut toujours vérifier qu'une fonction n'existe pas avant de l'écrire.

9. Les tableaux

9.1. Les tableaux numérotés

Un tableau (aussi appelé array) est une variable qui permet d'enregistrer de nombreuses informations. On en distingue deux types :

- les tableaux numérotés ;
- les tableaux associatifs.

Les tableaux numérotés associent des clés indexées à des valeurs.

Clé	**Valeur**
0	François
1	Michel
2	Nicole

Un array numéroté commence toujours à l'index n°0 !

```
<?php
//---- La fonction array permet de créer un tableau
$prenoms = array ('François', 'Michel', 'Nicole");

//---- on peut également créer manuellement le tableau :
$prenom = array() ;             // déclaration du tableau
$prenoms[0] = 'François';       // affectation élément par élément
$prenoms[1] = 'Michel';
$prenoms[2] = 'Nicole';

//---- PHP incrémente automatiquement la clé en laissant les crochets vides :
$prenoms[] = 'François';        // Créera $prenoms[0]
$prenoms[] = 'Michel';          // Créera $prenoms[1]
$prenoms[] = 'Nicole';          // Créera $prenoms[2]
?>
```

Pour afficher un élément, il faut donner sa position entre crochets après $prenoms.

```
<?php
// on parcourt le tableau de façon itérative, avec for
```

```
// la fonction count renvoie le nombre d'éléments dans le tableau
for ($i = 0 ; $i < count($prenom) ; $i++)
{
	echo "index $i : $prenoms[$i]<br/>";
}

// 2ème méthode (sans utiliser l'index), avec foreach
// foreach parcourt le tableau $prenom et stoke la valeur trouvée dans $value
foreach ($prenom as $value)
{
	echo "prénom : $value<br/>";
}
?>
```

9.2. Les tableaux associatifs

Les tableaux associatifs fonctionnent sur le même principe, sauf qu'au lieu de numéroter les index, on va les étiqueter en leur donnant à chacune un nom différent.

Les tableaux numérotés permettent de stocker une série d'éléments du même type, comme des prénoms alors que les tableaux associatifs permettent de découper une donnée en plusieurs sous-éléments.

```
<?php
//---- La fonction array permet de créer un tableau
// indiquer une flèche (=>) pour dire « associé à »
$coordonnees = array (
	'prenom' => 'François',
	'nom' => 'Dupont',
	'adresse' => '3 Rue du Paradis',
	'ville' => 'Marseille'
	);

//---- on peut également créer manuellement le tableau :
$coordonnees['prenom'] = 'François';
$coordonnees['nom'] = 'Dupont';
$coordonnees['adresse'] = '3 Rue du Paradis';
$coordonnees['ville'] = 'Marseille';
?>
```

Pour afficher un élément, il suffit d'indiquer le nom de cet élément entre crochets, ainsi qu'entre guillemets ou apostrophes puisque l'étiquette du tableau associatif est un texte.

```
<?php
// affichage du prénom et du nom
echo $coordonnees['prenom'] .' '. $coordonnees['nom'];

//---- parcours du tableau pour récupérer la valeur
foreach ($coordonnees as $valeur)
{
```

```
	echo $valeur .'<br/>';
}

//---- parcours du tableau pour récupérer la valeur et la clé
foreach ($coordonnees as $cle => $valeur)
{
	echo $cle .' : '.  $valeur .'<br/>';
}
?>
```

9.3. Recherche dans un tableau

Trois types de recherches, basées sur des fonctions PHP :

- array_key_exists : pour vérifier si une clé existe dans l'array ;
- in_array : pour vérifier si une valeur existe dans l'array ;
- array_search : pour récupérer la clé d'une valeur dans l'array.

```
<?php
// initialisation du tableau
$coordonnees = array (
	'prenom' => 'François',
	'nom' => 'Dupont',
	'adresse' => '3 Rue du Paradis',
	'ville' => 'Marseille'
	);

// La fonction array_key_exists($cle, $array) renvoie un booléen
// true (vrai) si la clé est trouvée ou false (faux) si la clé ne s'y trouve pas.
if ( array_key_exists('pays', $coordonnees) == false )
{
	echo 'La clé "pays" ne se trouve dans les coordonnées !<br/>';
}
else
{
	// La fonction in_array($valeur, $array) renvoie un booléen
	// true (vrai) si la valeur est trouvée ou false (faux) si elle ne s'y trouve pas.
	if ( in_array('Paris', $coordonnees) )
	{
		echo 'Une personne habite Paris.<br/>';

		// La fonction array_search($valeur, $array) renvoie la clé correspondante :
		// → le numéro si c'est un tableau numéroté
		// → ou le nom de la clé si c'est un array associatif
		// NB : la fonction renvoie false si elle n'a pas trouvé la valeur
		$position = array_search('Paris', $coordonnees) ;
		echo '"Paris" se trouve en position ' . $position . '<br />';
	}
```

```
}
?>
```

9.4. Tableaux multi dimensionnels

Un tableau multidimensionnel n'est rien d'autre qu'un tableau contenant au minimum deux tableaux. Il peut être numéroté ou associatif.

Chaque dimension est désignée par des crochets [].

```
<?php
// initialisation tableau numéroté
// une chaîne de caractères (string) n'est rien d'autre qu'un tableau de lettres
$prenoms = array ('François', 'Michel', 'Nicole');

// on peut ainsi accéder à chacune des lettres d'un élément de tableau
// ex : afficher toutes les lettres du 2ème prénom
for ($i=0; $i < strlen($prenoms[1]); $i++)
{
	// NB : la fonction strlen renvoie la longueur d'une chaîne de caractères
	print($prenoms[1][$i] ."<br/>");
}

//initialisation tableau associatif
$tab = array (
	0=>array("nom"=>"bastien", "prenom"=>"jean", "age"=>25),
	1=>array("nom"=>"lola", "prenom"=>"gerard", "age"=>35),
	2=>array("nom"=>"joseph","prenom"=>"mahel", "age"=>58)
	);

foreach($tab as $cle => $valeur)
{
	print($valeur['nom']."<br/>");
}
?>
```

TEST

1 - Vrai ou faux ? Un navigateur web peut exécuter et lire du code PHP.

- Vrai
- Faux

2 - Laquelle de ces balises délimite du code PHP ?

- <?php ?>
- <php /php>
- <? php?>
- <!php ?>

3 - Les lignes de code PHP sont appelées des instructions. Par quel symbole se terminent-elles ?

- Un point
- Un point-virgule
- Un dièse
- Il n'y a aucun symbole particulier

4 - Quels logiciels sont inclus dans WAMP ?

- Apache, PHP, Winamp
- PHP, Apache, Chrome
- PHP, Apache, MySQL
- Comanche, Tango, Vaudou

5 - Une même page peut-elle être incluse dans plusieurs pages différentes d'un site web grâce à PHP ?

- Oui, grâce à include
- Oui, c'est automatique si les fichiers sont dans le même dossier
- Non

6 - Laquelle de ces valeurs est un booléen ?

- 13
- "Bonjour"
- false
- 0.5

7 - Lequel de ces mots-clés indique à l'ordinateur de répéter des instructions ?

- for
- if
- select
- switch

8 - Que renvoie la fonction strlen de PHP ?

- La longueur d'un texte
- Rien du tout
- Une version mélangée d'un texte
- Les premiers mots d'un texte

9 - Les tableaux PHP sont par défaut numérotés à partir de...

- L'index 0
- L'index 1
- L'index 2

10 - Laquelle de ces conditions dit : "Si le prix est strictement supérieur à 10" ?

- if ($prix = 10)
- if ($prix < 10)
- if ($prix >= 10)
- if ($prix > 10)