Langage PHP

Table des matières

[Base 2](#_Toc435714706)

[Les Variable 3](#_Toc435714707)

[Les conditions 4](#_Toc435714708)

[Boucles 5](#_Toc435714709)

[Fonctions 6](#_Toc435714710)

[Les Tableaux 9](#_Toc435714711)

[Securitee URL 15](#_Toc435714712)

# Base

* On considère qu'il existe deux types de sites web : les sites **statiques** et les sites **dynamiques.**
* **Les sites statiques** : ce sont des sites réalisés uniquement à l'aide des langages HTML et CSS. Ils fonctionnent très bien mais leur contenu ne peut pas être mis à jour automatiquement : il faut que le propriétaire du site (le webmaster) modifie le code source pour y ajouter des nouveautés. Ce n'est pas très pratique quand on doit mettre à jour son site plusieurs fois dans la même journée ! Les sites statiques sont donc bien adaptés pour réaliser des sites « vitrine », pour présenter par exemple son entreprise, mais sans aller plus loin. Ce type de site se fait de plus en plus rare aujourd'hui, car dès que l'on rajoute un élément d'interaction (comme un formulaire de contact), on ne parle plus de site statique mais de site dynamique.
* **Les sites dynamiques** : plus complexes, ils utilisent d'autres langages en plus de HTML et CSS, tels que PHP et MySQL. Le contenu de ces sites web est dit « dynamique » parce qu'il peut changer sans l'intervention du webmaster ! La plupart des sites web que vous visitez aujourd'hui, y compris le Site du Zéro, sont des sites dynamiques. Le seul prérequis pour apprendre à créer ce type de sites est de déjà savoir réaliser des sites statiques en HTML et CSS.

Une balise PHP elle commence par

 <?php  et se termine par ?>  ; c'est à l'intérieur que l'on mettra du code PHP, ce que je vais vous apprendre tout au long de ce cours.

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

Il existe d'autres balises pour utiliser du PHP, par exemple <? ?>, <% %>, etc. Ne soyez donc pas étonnés si vous en voyez. Néanmoins, <?php ?> est la forme la plus correcte, vous apprendrez donc à vous servir de cette balise et non pas des autres.

En résumé

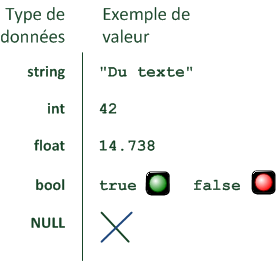
#### Les instructions PHP sont placées dans une balise**<?php ?> .**

#### Pour afficher du texte en PHP, on utilise l'instruction**echo .**

#### Ajouter des commentaires en PHP. On utilise pour cela les symboles**//**ou**/\* \*/ .**

* L'instruction include sera remplacée par le contenu de la page demandée.

# Les Variable



* On écrit le symbole « dollar » ($) : il précède toujours le nom d'une variable. Vous reconnaîtrez toujours qu'il y a une variable par la présence du symbole « dollar » ($).
* Ensuite, il y a le signe « égal » (=) : c'est pour dire que $age\_du\_visiteur est égal à…
* À la suite, il y a la valeur de la variable
* Enfin, il y a l'incontournable point-virgule (;) qui permet de terminer l'instruction.
* Entourer votre texte de guillemets doubles "" ou de guillemets simples

# Les conditions

* Les conditions permettent à PHP de prendre des décisions en fonction de la valeur des variables.
* La forme de condition la plus courante est if… elseif… else qui signifie « si »… « sinon si »… « sinon ».
* On peut combiner des conditions avec les mots-clés AND (« et ») et OR (« ou »).
* Si une condition comporte de nombreux elseif, il peut être plus pratique d'utiliser switch, une autre forme de condition.
* Les ternaires sont des conditions. Elles sont cependant plus rarement utilisées.

while peut se traduire par TANT QUE**$continuer\_boucle**est vrai, exécuter ces instructions.

<?php

$nombre\_de\_lignes = 1;

while ($nombre\_de\_lignes <= 100)

{

echo 'Je ne dois pas regarder les mouches voler quand j\'apprends le PHP.<br />';

$nombre\_de\_lignes++; // $nombre\_de\_lignes = $nombre\_de\_lignes + 1

}

?>

La particularité, là, c'est qu'on affiche à chaque fois la valeur de $nombre\_de\_lignes (ça vous permet de voir que sa valeur augmente petit à petit).

Exemple avec switch

<?php

$note = 10;

switch ($note) // on indique sur quelle variable on travaille

{

case 0: // dans le cas où $note vaut 0

echo "Tu es vraiment un gros nul !!!";

break;

case 5: // dans le cas où $note vaut 5

echo "Tu es très mauvais";

break;

case 7: // dans le cas où $note vaut 7

echo "Tu es mauvais";

break;

. . .}

?>

# Boucles

* Les deux principaux types de boucles sont :
  + while : à utiliser de préférence lorsqu'on ne sait pas par avance combien de fois la boucle doit être répétée ;
* for : à utiliser lorsqu'on veut répéter des instructions un nombre précis de fois.
* L'incrémentation est une technique qui consiste à ajouter 1 à la valeur d'une variable.
* La décrémentation retire au contraire 1 à cette variable. On trouve souvent des incrémentations au sein de boucles for.
* <?php
* $nombre\_de\_lignes = 1;
* while ($nombre\_de\_lignes <= 100)
* {
* echo 'Ceci est la ligne n°' . $nombre\_de\_lignes . '<br />';
* $nombre\_de\_lignes++;
* }
* ?>
* cette fois avec un for :
* <?php
* for ($nombre\_de\_lignes = 1; $nombre\_de\_lignes <= 100; $nombre\_de\_lignes++)
* {
* echo 'Ceci est la ligne n°' . $nombre\_de\_lignes . '<br />';
* }
* ?>

# Fonctions

***Exemple de fonction***

##### **strlen**: Cette fonction retourne la longueur d'une chaîne de caractères, c'est-à-dire le nombre de lettres et de chiffres dont elle est constituée (espaces compris).Exemple :

<?php

$phrase = 'Bonjour tout le monde ! Je suis une phrase !';

$longueur = strlen($phrase);

echo 'La phrase ci-dessous comporte ' . $longueur . ' caractères :<br ></code>' . $phrase;

?>

Résultat : Un comptage du nombre de caractères

str\_replace remplace une chaîne de caractères par une autre. Exemple :

<?php

$ma\_variable = str\_replace('b', 'p', 'bim bam boum');

echo $ma\_variable;

?>

Résultat : Remplacement de caractères

##### str\_shuffle :Pour vous amuser à mélanger aléatoirement les caractères de votre chaîne !

<?php

$chaine = 'Cette chaîne va être mélangée !';

$chaine = str\_shuffle($chaine);

echo $chaine;

?>

Résultat : Lettres mélangées

strtolower met tous les caractères d'une chaîne en minuscules.

<?php

$chaine = 'COMMENT CA JE CRIE TROP FORT ???';

$chaine = strtolower($chaine);

echo $chaine;

?>

Résultat : Lettres en minuscules

À noter qu'il existe strtoupper qui fait la même chose en sens inverse :

minuscules → majuscules.

Récupérer la date

Il s'agit de date Cette fonction peut donner beaucoup d'informations. Voici les principaux paramètres à connaître :

| **Paramètre** | **Description** |
| --- | --- |
| H | Heure |
| i | Minute |
| d | Jour |
| m | Mois |
| Y | Année |

Attention ! Respectez les majuscules/minuscules, c'est important !

<?php

$annee = date('Y');

echo $annee;

?>

On peut bien entendu faire mieux, voici la date complète et l'heure :

<?php

// Enregistrons les informations de date dans des variables

$jour = date('d');

$mois = date('m');

$annee = date('Y');

$heure = date('H');

$minute = date('i');

// Maintenant on peut afficher ce qu'on a recueilli

echo 'Bonjour ! Nous sommes le ' . $jour . '/' . $mois . '/' . $annee . 'et il est ' . $heure. ' h ' . $minute;

?>

Résultat : Affichage de la date

Pour créer une fonction, vous devez taper function

<?php

function DireBonjour($nom)

{

echo 'Bonjour ' . $nom . ' !<br ></code>';

}

DireBonjour('Marie');

DireBonjour('Patrice');

DireBonjour('Edouard');

DireBonjour('Pascale');

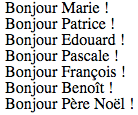
DireBonjour('François');

DireBonjour('Benoît');

DireBonjour('Père Noël');

?>

Résultat :



# Les Tableaux

Ici, nous allons voir qu'il est possible d'enregistrer de nombreuses informations dans une seule variable grâce aux tableaux. On en distingue deux types :

les tableaux numérotés ;

les tableaux associatifs.

Pour créer un tableau numéroté en PHP, on utilise généralement la fonction array.

Cet exemple vous montre comment créer l'array $prenoms :

<?php

// La fonction array permet de créer un array

$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');

?>

L'ordre a beaucoup d'importance. Le premier élément (« François ») aura le n°0, ensuite Michel le n°1, etc.

Vous pouvez aussi créer manuellement le tableau case par case :

<?php

$prenoms[0] = 'François';

$prenoms[1] = 'Michel';

$prenoms[2] = 'Nicole';

?>

Si vous ne voulez pas avoir à écrire vous-mêmes le numéro de la case que vous créez, vous pouvez laisser PHP le sélectionner automatiquement en laissant les crochets vides :

<?php

$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

echo $prenoms[1];

?>

Les tableaux associatifs

Pour en créer un, on utilisera la fonction array , mais on va mettre « l'étiquette » devant chaque information :

<?php

// On crée notre array $coordonnees

$coordonnees = array (

'prenom' => 'François',

'nom' => 'Dupont',

'adresse' => '3 Rue du Paradis',

'ville' => 'Marseille');

?>

écrit une flèche (=>) pour dire « associé à ».On dit « *ville* » associée à « *Marseille* ».

Il est aussi possible de créer le tableau case par case, comme ceci :

<?php

$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

echo $coordonnees['ville'];

?>

Lorsqu'un tableau a été créé, on a souvent besoin de le parcourir pour savoir ce qu'il contient. Nous allons voir trois moyens d'explorer un array :

* la boucle for ;
* la boucle foreach ;
* la fonction print\_r (utilisée principalement pour le débogage).

#### La boucle for

Il est très simple de parcourir un tableau numéroté avec une boucle for. En effet, puisqu'il est numéroté à partir de 0, on peut faire une boucle for qui incrémente un compteur à partir de 0 :

<?php

// On crée notre array $prenoms

$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');

// Puis on fait une boucle pour tout afficher :

for ($numero = 0; $numero < 5; $numero++)

{

echo $prenoms[$numero] . '<br />'; // affichera $prenoms[0], $prenoms[1] etc.

}

?>



#### La boucle foreach

La boucle for a beau fonctionner, on peut utiliser un autre type de boucle plus adapté aux tableaux qu'on n'a pas encore vu jusqu'ici : foreach. C'est une sorte de boucle for spécialisée dans les tableaux.

foreach va passer en revue chaque ligne du tableau, et lors de chaque passage, elle va mettre la valeur de cette ligne dans une variable temporaire (par exemple $element).

<?php

$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');

foreach($prenoms as $element)

{

echo $element . '<br />'; // affichera $prenoms[0], $prenoms[1] etc.

}

?>



L'avantage de foreach est qu'il permet aussi de parcourir les tableaux associatifs.

<?php

$coordonnees = array (

'prenom' => 'François',

'nom' => 'Dupont',

'adresse' => '3 Rue du Paradis',

'ville' => 'Marseille');

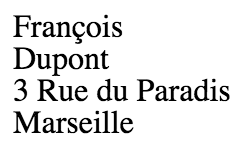
foreach($coordonnees as $element)

{

echo $element . '<br />';

}

?>



On peut aussi récupérer la clé de l'élément. On doit dans ce cas écrire foreach comme ceci :

<?php foreach($coordonnees as $cle => $element) ?>

À chaque tour de boucle, on disposera non pas d'une, mais de deux variables :

* $cle, qui contiendra la clé de l'élément en cours d'analyse (« prenom », « nom », etc.) ;
* $element, qui contiendra la valeur de l'élément en cours (« François », « Dupont », etc.).

<?php

$coordonnees = array (

'prenom' => 'François',

'nom' => 'Dupont',

'adresse' => '3 Rue du Paradis',

'ville' => 'Marseille');

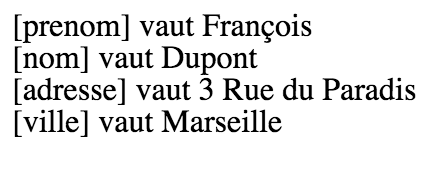
foreach($coordonnees as $cle => $element)

{

echo '[' . $cle . '] vaut ' . $element . '<br />';

}

?>



#### Afficher rapidement un array avec print\_r

Parfois, en codant votre site en PHP, vous aurez sous les bras un array et vous voudrez savoir ce qu'il contient, juste pour votre information savoir ce que contient l'array, vous pouvez faire appel à la fonction print\_r. C'est une sorte de  echo spécialisé dans les arrays.

Pour bien les voir, il faut donc utiliser la balise HTML <pre> qui nous permet d'avoir un affichage plus correct.

<?php

$coordonnees = array (

'prenom' => 'François',

'nom' => 'Dupont',

'adresse' => '3 Rue du Paradis',

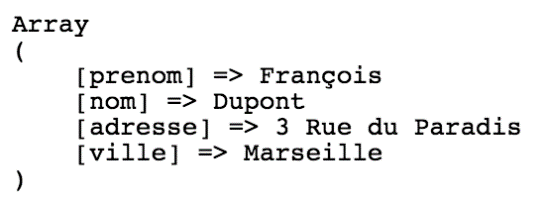
'ville' => 'Marseille');

echo '<pre>';

print\_r($coordonnees);

echo '</pre>';

?>



Nous allons voir 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.

On doit d'abord lui donner le nom de la clé à rechercher, puis le nom de l'array dans lequel on fait la recherche :

<?php array\_key\_exists('cle', $array); ?>

La fonction renvoie un booléen, c'est-à-dire true (vrai) si la clé est dans l'array, et false (faux) si la clé ne s'y trouve pas. Ça nous permet de faire un test facilement avec un if :

<?php

$coordonnees = array (

'prenom' => 'François',

'nom' => 'Dupont',

'adresse' => '3 Rue du Paradis',

'ville' => 'Marseille');

if (array\_key\_exists('nom', $coordonnees))

{

echo 'La clé "nom" se trouve dans les coordonnées !';

}

if (array\_key\_exists('pays', $coordonnees))

{

echo 'La clé "pays" se trouve dans les coordonnées !';

}

?>

https://sdz-upload.s3.amazonaws.com/prod/upload/0149.png

On recherche dans les valeurs. in\_array renvoie true si la valeur se trouve dans l'array, false si elle ne s'y trouve pas.

Pour changer un peu de notre array $coordonnees, je vais en créer un nouveau (numéroté) composé de fruits. ;-)

<?php

$fruits = array ('Banane', 'Pomme', 'Poire', 'Cerise', 'Fraise', 'Framboise');

if (in\_array('Myrtille', $fruits))

{

echo 'La valeur "Myrtille" se trouve dans les fruits !';

}

if (in\_array('Cerise', $fruits))

{

echo 'La valeur "Cerise" se trouve dans les fruits !';

}

?>

Vérifier si la clé se trouve dans l'array $fruits

#### Récupérer la clé d'une valeur dans l'array : array\_search

array\_search fonctionne comme in\_array : il travaille sur les valeurs d'un array. Voici ce que renvoie la fonction :

* si elle a trouvé la valeur, array\_search renvoie la clé correspondante (c'est-à-dire le numéro si c'est un array *numéroté*, ou le nom de la clé si c'est un array *associatif*);
* si elle n'a pas trouvé la valeur, array\_search renvoie false.

On reprend l'array numéroté avec les fruits :

<?php

$fruits = array ('Banane', 'Pomme', 'Poire', 'Cerise', 'Fraise', 'Framboise');

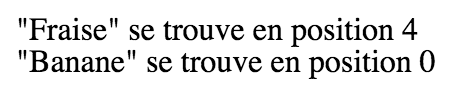
$position = array\_search('Fraise', $fruits);

echo '"Fraise" se trouve en position ' . $position . '<br />';

$position = array\_search('Banane', $fruits);

echo '"Banane" se trouve en position ' . $position;

?>



# Securitee URL

#### En résumé

* Une URL représente l'adresse d'une page web (commençant généralement par http://).
* Lorsqu'on fait un lien vers une page, il est possible d'ajouter des paramètres sous la forme bonjour.php?nom=Dupont&prenom=Jean qui seront transmis à la page.
* La page bonjour.php dans l'exemple précédent recevra ces paramètres dans un array nommé $\_GET :
  + $\_GET['nom'] aura pour valeur « Dupont » ;
  + $\_GET['prenom'] aura pour valeur « Jean ».
* Cette technique est très pratique pour transmettre des valeurs à une page, mais il faut prendre garde au fait que le visiteur peut les modifier très facilement. Il ne faut donc pas faire aveuglément confiance à ces informations, et tester prudemment leur valeur avant de les utiliser.
* La fonction isset() permet de vérifier si une variable est définie ou non.
* Le transtypage est une technique qui permet de convertir une variable dans le type de données souhaité. Cela permet de s'assurer par exemple qu'une variable est bien un int (nombre entier).

## **TRANSMETTRE DES DONNÉES AVEC LES FORMULAIRES**

En HTML, pour insérer un formulaire, on se sert de la balise <form>. On l'utilise de la manière suivante :

<form method="post" action="cible.php">

<p>

On insèrera ici les éléments de notre formulaire.

</p>

</form>

#### La méthode

Il faut savoir qu'il existe plusieurs moyens d'envoyer les données du formulaire (plusieurs « méthodes »). Vous pouvez en employer deux.

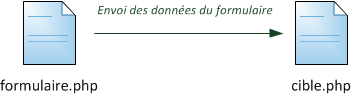
* get : les données transiteront par l'URL comme on l'a appris précédemment. On pourra les récupérer grâce à l'array $\_GET. Cette méthode est assez peu utilisée car on ne peut pas envoyer beaucoup d'informations dans l'URL (je vous disais dans le chapitre précédent qu'il était préférable de ne pas dépasser 256 caractères).
* post : les données ne transiteront pas par l'URL, l'utilisateur ne les verra donc pas passer dans la barre d'adresse. Cette méthode permet d'envoyer autant de données que l'on veut, ce qui fait qu'on la privilégie le plus souvent. Néanmoins, les données ne sont pas plus sécurisées qu'avec la méthode GET et il faudra toujours vérifier si tous les paramètres sont bien présents et valides, comme on l'a fait dans le chapitre précédent. **On ne doit pas plus faire confiance aux formulaires qu'aux URL**.

C'est à vous de choisir par quelle méthode vous souhaitez que les données du formulaire soient envoyées. Si vous hésitez, sachez que dans 99 % des cas la méthode que l'on utilise est post, vous écrirez doncmethod="post" comme je l'ai fait.

#### La cible

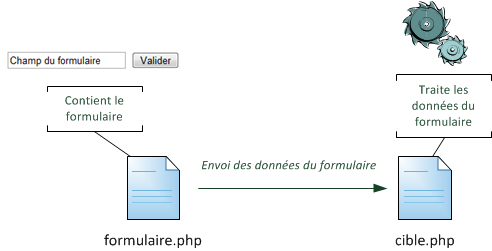
L'attribut action sert à définir la page appelée par le formulaire. C'est cette page qui recevra les données du formulaire et qui sera chargée de les traiter.

Imaginons le schéma de la figure suivante.



Appel de la page cible par le formulaire

Dans cet exemple, le formulaire se trouve dans la page formulaire.php. Cette page ne fait aucun traitement particulier, mais une fois le formulaire envoyé (lorsqu'on a cliqué sur le bouton « Valider »), le visiteur est redirigé vers la page cible.php qui reçoit les données du formulaire, comme vous le montre la figure suivante.

Fonctionnement d'un formulaire

Le nom de la page cible est défini grâce à l'attribut action.

Retenez donc bien que vous travaillez normalement sur deux pages différentes : la page qui contient le formulaire (formulaire.php dans notre exemple), et celle qui reçoit les données du formulaire pour les traiter (cible.php).

### **Les éléments du formulaire**

<https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql/transmettre-des-donnees-avec-les-formulaires#/id/r-912918>

#### En résumé

* Les formulaires sont le moyen le plus pratique pour le visiteur de transmettre des informations à votre site. PHP est capable de récupérer les données saisies par vos visiteurs et de les traiter.
* Les données envoyées via un formulaire se retrouvent dans un array $\_POST.
* De la même manière que pour les URL, il ne faut pas donner sa confiance absolue aux données que vous envoie l'utilisateur. Il pourrait très bien ne pas remplir tous les champs voire trafiquer le code HTML de la page pour supprimer ou ajouter des champs. Traitez les données avec vigilance.
* Que ce soit pour des données issues de l'URL ($\_GET) ou d'un formulaire ($\_POST), il faut s'assurer qu'aucun texte qui vous est envoyé ne contient du HTML si celui-ci est destiné à être affiché sur une page. Sinon, vous ouvrez une faille appelée XSS qui peut être néfaste pour la sécurité de votre site.
* Pour éviter la faille XSS, il suffit d'appliquer la fonction htmlspecialchars sur tous les textes envoyés par vos visiteurs que vous afficherez.
* Les formulaires permettent d'envoyer des fichiers. On retrouve les informations sur les fichiers envoyés dans un array $\_FILES. Leur traitement est cependant plus complexe.

## **VARIABLES SUPERGLOBALES, SESSIONS ET COOKIES**

#### <https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql/variables-superglobales-sessions-et-cookies>

#### En résumé

* Les variables superglobales sont des variables automatiquement créées par PHP. Elles se présentent sous la forme d'arrays contenant différents types d'informations.
* Dans les chapitres précédents, nous avons découvert deux superglobales essentielles : $\_GET (qui contient les données issues de l'URL) et $\_POST (qui contient les données issues d'un formulaire).
* La superglobale $\_SESSION permet de stocker des informations qui seront automatiquement transmises de page en page pendant toute la durée de visite d'un internaute sur votre site. Il faut au préalable activer les sessions en appelant la fonction session\_start().
* La superglobale $\_COOKIE représente le contenu de tous les cookies stockés par votre site sur l'ordinateur du visiteur. Les cookies sont de petits fichiers que l'on peut écrire sur la machine du visiteur pour retenir par exemple son nom. On crée un cookie avec la fonction setcookie().

## **LIRE ET ÉCRIRE DANS UN FICHIER**

<https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql/lire-et-ecrire-dans-un-fichier-5>

#### En résumé

* PHP permet d'enregistrer des informations dans des fichiers sur le serveur.
* Il faut au préalable s'assurer que les fichiers autorisent PHP à les modifier. Pour cela, il faut changer les permissions du fichier (on parle de CHMOD) à l'aide d'un logiciel FTP comme FileZilla. Donnez la permission 777 au fichier pour permettre à PHP de travailler dessus.
* La fonction fopen permet d'ouvrir le fichier, fgets de le lire ligne par ligne et fputs d'y écrire une ligne.
* À moins de stocker des données très simples, l'utilisation des fichiers n'est pas vraiment la technique la plus adaptée pour enregistrer des informations. Il est vivement recommandé de faire appel à une base de données.

FIN 2 PARTIE