



Chapitre

Le langage P.H.P.

PHP

V5

PHP est un langage de programmation créé par Ramsus Lerdorf (1994).



C'est un puissant langage de script embarqué dans les pages HTML et traité par le serveur, pour dynamiser les pages Web.
C'est un logiciel « libre » (gratuit et « open source »).

PHP est un module supporté entre autre par le serveur web Apache, le plus répandu dans le monde, il est donc développé pour être facilement utilisable via ce serveur (Il fonctionne évidemment avec d'autres serveurs web). PHP permet d'interfacer très facilement de très nombreuses bases de données notamment MySQL : gratuite et performante.

LE LANGAGE P.H.P.

1°) Qu'est ce que PHP?

Officiellement « P.H.P. : Préprocesseur HyPertexte » .

C' est un langage de script , qui fonctionne coté serveur.

Ce qui distingue le PHP des autres langages de script comme le Javascript c' est que le code est exécuté sur le serveur.

Si vous avez un script similaire sur votre serveur, le client ne reçoit que le résultat du script, sans aucun moyen d'avoir accès au code qui a produit ce résultat.

1°) Le balisage du code PHP

Le code PHP est inclus entre une balise de début et une balise de fin qui permettent au navigateur de passer en "mode PHP". Le code PHP doit être écrit dans la zone **Body** du fichier HTML.

LE LANGAGE P.H.P.

méthode 1:

```
<?php  
    echo("Si vous voulez afficher du XML ou du XHTML, faites comme ceci.");  
?>
```

méthode 2: (la plus complète)

```
<script language="php">  
    echo ("Certains éditeurs HTML n'acceptent pas les expressions telles que celle-ci.");  
</script>
```

LE LANGAGE P.H.P.

2°) Les règles de syntaxe du PHP

Les instructions doivent se terminer par un point-virgule « ; » comme en C

Les commentaires sont identiques au C :

// commentaire pour une ligne

*/**

commentaire sur plusieurs lignes

**/*

commentaire comme en Shell Unix

Il est possible de morceler le code PHP pour y mêler du code HTML :

```
<?php
    if ( $i==1 )
    {
?>        <strong>Ceci est vrai.</strong>
<?php    } else {
?>        <strong>Ceci est faux.</strong>
<?php    }
?>
```

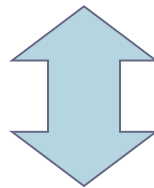
Diagram illustrating the interleaving of PHP and HTML code:

- Red double-headed arrows indicate blocks of **Code PHP**:
 - Between the opening `<?php` and the first `?>`.
 - Between the opening `<?php` and the closing `?>` of the first PHP block.
 - Between the opening `<?php` and the closing `?>` of the second PHP block.
 - Between the opening `<?php` and the closing `?>` of the third PHP block.
- Red double-headed arrows indicate blocks of **Code html**:
 - Between the first `?>` and the opening `<?php` of the second PHP block.
 - Between the second `?>` and the opening `<?php` of the third PHP block.

LE LANGAGE P.H.P.

Exemple de code PHP : ATTENTION le programme étant avec du Php doit se terminer par **.php**

```
<html>
  <head>
    <title>Exemple</title>
  </head>
  <body>
    <?php
      echo "Bonjour, je suis un script PHP!";
    ?>
    <h1>Ceci est un <?php echo "simple";?>exemple.</h1><p>
  </body>
</html>
```



LE LANGAGE P.H.P.

II°) LES TYPES DE DONNEES-LES VARIABLES

1°) Généralités sur les types de donnée en PHP

Le PHP connaît les types de données suivant :



- booléen
- entier (décimal, octal, hexadécimal)
- nombre à virgule flottante (format scientifique ou non, séparateur: .)
- chaîne de caractères

PHP supporte deux types composés :

- tableau
- objet

PHP supporte le type spécial :



- null : représente l'absence de valeur. Une variable avec la valeur NULL n'a pas de valeur.

Le type de variable est déterminé par le contexte de la variable sans que le programmeur ait besoin de la définir explicitement.



: signifie que cela ne fonctionne qu' à partir de PHP4
(C) PLAG

LE LANGAGE P.H.P.

2°) Le nommage des variables

En PHP, les variables sont représentées par un signe dollar "\$" suivi du nom de la variable.

Le nom est sensible aux majuscules (\$x != \$X).

Les noms de variables suivent les mêmes règles de nommage que les autres entités PHP.

Un nom de variable valide doit commencer par une lettre ou un souligné (_), suivi de lettres, chiffres ou soulignés.

Exemples

```
<?php
$var = "Jean";
$Var = "Paul";           // Attention à la majuscule Var!=var
$_4site = 'pas encore';
$maïs = 'jaune';         // valide; 'i' est ASCII 239.
?>
```

Note : une lettre peut être de (a à z) ou de (A à Z), et les caractères ASCII de 127 à 255

LE LANGAGE P.H.P.

3°) Utilisation des types

Pour les booléens :

true => 1 et false=>0

```
<?php
```

```
$variable1 = true;
```

```
$variable2 = false;
```

```
?>
```

Pour les entiers :

entier signé de 32 bits

```
<?php
```

```
$variable1 = 523;
```

```
$variable2 = -523;
```

```
$variable3 = 01013; // valeur en octal car un « 0 » devant le chiffre => (523)10
```

```
$variable4 = 0x20B; // valeur en hexadécimal car « 0x » devant le chiffre => (523)10
```

```
?>
```

Pour les nombres flottants :

pas de limite réelle

```
<?php
```

```
$variable1 = 3.1456; // notation standard
```

```
$variable2 = 0.31456e1; // notation scientifique
```

```
?>
```

LE LANGAGE P.H.P.

Pour les chaînes de caractères : **pas de limite de taille**

Guillemets simples : 'bonjour'

dans cette représentation aucun caractère de format à l'affichage ne marche

exemple : *echo 'Etes vous sur de vouloir effacer le dossier C:*.?*';*

Guillemets doubles : "bonjour"

Si la chaîne est entourée de guillemets doubles, PHP va accepter certaines séquences de caractères :

Séquence	Valeur
<code>\n</code>	Nouvelle ligne (linefeed, LF ou 0x0A (10) en ASCII)
<code>\r</code>	Retour à la ligne(carriage return, CR ou 0x0D (13) en ASCII)
<code>\t</code>	Tabulation horizontale (HT ou 0x09 (9) en ASCII)
<code>\\</code>	Antislash
<code>\\$</code>	Caractère \$
<code>\"</code>	Guillemets doubles
<code>\[0-7]{1,3}</code>	Une séquence de caractères qui permet de rechercher un nombre en notation octale.
<code>\x[0-9A-Fa-f]{1,2}</code>	Une séquence de caractères qui permet de rechercher un nombre en notation hexadécimale.

exemple : *echo "\n\rEtes vous sur de vouloir effacer le dossier C:*.?*";*

LE LANGAGE P.H.P.

4°) Traitement des variables dans les chaînes

Lorsqu'une chaîne est spécifiée avec des guillemets doubles, les variables qu'elle contient sont remplacées par leur valeur.

Il y a deux types de syntaxe :

- la simple : elle fournit un moyen d'utiliser les variables, que ce soit des chaînes, des tableaux ou des membres d'objets.

exemple :

<?php

\$boisson = 'vin';

echo "Du \$boisson, du pain et du fromage! ";

?>

- la complexe : car elle permet l'utilisation d'expressions complexes.

Il suffit d'écrire une expression exactement comme si elle était hors de la chaîne, puis de l'entourer d'accolades {}.

exemple :

<?php

\$super = 'fantastique';

echo "C'est {\$super}";

?>

LE LANGAGE P.H.P.

5°) Accès aux caractères d'une chaîne

Les caractères d'une chaîne sont accessibles en spécifiant leur offset (le premier caractère est d'offset 0) entre accolade, après le nom de la variable.

Exemple :

```
<?php
$str = "Ceci est une chaîne";
$str = $str . " avec un peu plus de texte";
$str .= " et une nouvelle ligne à la fin.\n";
$first = $str{0};
$last = $str{strlen($str)-1};
//.....
?>
```

/ Une méthode de concaténation*/
/* autre méthode de concaténation*/
/* Premier caractère d'une chaîne*/
/* Dernier caractère d'une chaîne*/*



Au format PHP3, exemple : *\$first = \$str[0]; // Syntaxe obsolète mais qui est pourtant la plus utilisée*

LE LANGAGE P.H.P.

6°) Conversion directe de type

Lorsqu'une chaîne de caractères est évaluée comme une valeur numérique, le résultat et le type de la variable sont déterminés comme suit :

- La chaîne de caractères est de type "double" si elle contient un des caractères '.', 'e' ou 'E'. Sinon, elle est de type entier ("integer").
- La valeur est définie par la première partie de la chaîne, si la chaîne de caractères débute par une valeur numérique cette valeur sera celle utilisée sinon, la valeur sera égale à 0 (zéro).
- Lorsque la première expression est une chaîne de caractères, le type de la variable dépend de la seconde expression.

Exemples :

<?php

```
$foo = 1 + "10.5";           // $foo est du type float (11.5)
$foo = 1 + "-1.3e3";         // $foo est du type float (-1299)
$foo = 1 + "bob-1.3e3";      // $foo est du type integer (1)
$foo = 1 + "bob3";           // $foo est du type integer (1)
$foo = 1 + "10 Small Pigs";  // $foo est du type integer (11)
$foo = 1 + "10 Little Piggies"; // $foo est du type integer (11)
$foo = "10.0 pigs " + 1;     // $foo est du type integer (11)
$foo = "10.0 pigs " + 1.0;   // $foo est du type float (11.0)
```

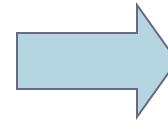
?>

LE LANGAGE P.H.P.

7°) Les variables externes à PHP - formulaires method POST

Lorsqu'un formulaire est envoyé vers un script PHP, tous les champs du formulaire seront disponibles dans le script sous forme d'une variable.

```
<html>
<head>    <title>formulaire.html</title>    </head>
<body>
<form action="form1.php" method="post">
    Votre nom: <input type="text" name="nom">
    <br>
    <input type="submit" value="Soumettre la requête">
</form>
</body>
</html>
```



ire 1.html

Votre nom:

Le PHP va faire passer le contenu de la saisie dans le champs *Name*: du formulaire au programme PHP nommé dans **Action**.

Dans l'exemple ci-dessus, la variable **\$nom** sera créée dans **form1.php** et contiendra la valeur du champs saisi par l'opérateur.

Le fait d'utiliser la méthode **post** permet de cacher la valeur des champs du formulaire à l'utilisateur

LE LANGAGE P.H.P.

`<html>`

`<head> <title>FORM1.PHP</title> </head>`

`<body>`

*récupération du champs nom de formulaire.htm
*

`<?php`

// Attribution du champs de formulaire à une variable locale au programme

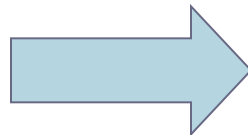
`$nom1=$_POST{'nom'}; // $_POST car method post dans le formulaire`

`echo "$nom1";`

`?>`

`</body>`

`</html>`



1.php

récupération du champs nom de formulaire.htm
Oscar

LE LANGAGE P.H.P.

Le PHP permet aussi l'utilisation des tableaux dans le contexte de formulaire, mais seulement des tableaux à une seule dimension. Comme cela, vous pouvez rassembler des variables ou utiliser cette fonctionnalité pour récupérer les valeurs d'un choix multiple

```
<form action="form2.php" method="post">  
    NOM:      <input type="text" name="perso"><br>  
    EMAIL:    <input type="text" name="mail"><br>  
<br>  
<select multiple name="vin{}"> ou <select multiple name=vin[<br>  
    <option value="medoc">Médoc<br>  
    <option value="chablis">Chablis<br>  
    <option value="riesling">Riesling<br>  
</select>  
<input type="submit"> </form>
```



NOM: LOLA
EMAIL: LOLA@XXX:XXX

Médoc
Chablis
Riesling

Envoyer

LE LANGAGE P.H.P.

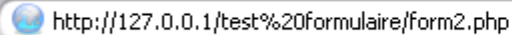
```
<html>
```

```
<head>    <title>Exemple</title>    </head>
```

```
<body>
```

*récupération du champs nom de formulaire.htm
*

```
<?php
```



```
$perso=$_POST{'perso'};
```

```
$mail=$_POST{'mail'};
```

```
$vin=$_POST{'vin'};
```

```
echo "$perso<br>";
```

```
echo "$mail<br>";
```

```
if (isset($vin{0})) echo "{$vin{0}}<br>";
```

```
if (isset($vin{1})) echo "{$vin{1}}<br>";
```

```
if (isset($vin{2})) echo "{$vin{2}}<br>";
```

```
?>
```

```
</body>
```

```
</html>
```



récupération du champs nom de formulaire.htm

LOLA

LOLA@XXX;XXX

medoc

riesling

NOTA :

La fonction Php **isset(nom de variable)** permet de tester si une variable existe et si elle contient des données. Cette fonction renvoie **true** si c'est vrai et **false** sinon.

LE LANGAGE P.H.P.

8°) Les variables externes à PHP - formulaires method GET

Le principe de fonctionnement est identique à la méthode précédente sauf que nous utilisons la method GET

```
<form action="form2.php" method="get">  
    NOM:    <input type="text" name="perso"><br>  
    EMAIL:  <input type="text" name="mail"><br>  
<br>  
<select multiple name="vin{}">  
    <option value="medoc">Médoc  
    <option value="chablis">Chablis  
    <option value="riesling">Riesling  
</select>  
<input type="submit"> </form>
```



NOM:

EMAIL:

LE LANGAGE P.H.P.

<html>

<head> <title>Exemple</title> </head>

<body>

récupération du champs nom de formulaire.htm

<?php

\$perso=\$_GET{'perso'};

\$mail=\$_GET{'mail'};

\$vin=\$_GET{'vin'};

*echo "\$perso
";*

*echo "\$mail
";*

*if (isset(\$vin{0})) echo "{ \$vin{0} }
";*

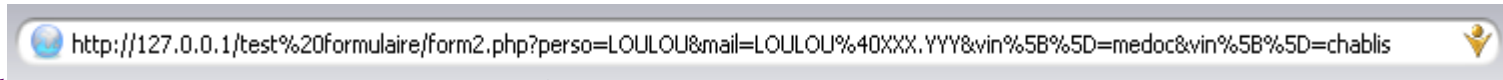
*if (isset(\$vin{1})) echo "{ \$vin{1} }
";*

*if (isset(\$vin{2})) echo "{ \$vin{2} }
";*

?>

</body>

</html>



récupération du champs nom de formulaire.htm
LOULOU
LOULOU@XXX.YYY
medoc
chablis

NOTA :

Dans la barre d' adresse du formulaire form2.php on voit bien l' ensemble des variables du formulaire et leurs contenus

LE LANGAGE P.H.P.


ALTERNATIVE A LA METHODE PRECEDENTE :

```
<?php
extract($_GET);      // récupère l'ensemble des variables du formulaire
                     // Marche aussi avec la méthode POST => extract($_POST)

echo "$perso<br>";
echo "$mail<br>";

if (isset($vin{0})) echo "{$vin{0}}<br>";
if (isset($vin{1})) echo "{$vin{1}}<br>";
if (isset($vin{2})) echo "{$vin{2}}<br>";

?>
```



Site d'aide PHP.NET