PHP

Développement

Jason Le Scour

Sommaire

Introduction

La programmation consiste, à réaliser un programme résolvant un problème donné ou satisfaisant à un besoin donné. Compte tenu de la multiplicité des langages existants, il existe différentes façon d'aborder cette activité.

Un programme c'est un ensemble fini d'opérations à réaliser et compris par une machine.

Le langage PHP

- Le sigle PHP signifie à l'origine *Personal Home Page.*
- Il s'agissait alors d'ajouter quelques fonctionnalités aux pages personnelles.
- Il permet de créer des pages web dynamique et interactives.
- PHP signifie aujourd'hui *PHP Hypertext* Pre-processor, car il renvoie à un navigateur un document HTML.
- PHP est un langage de script qui s'exécute coté serveur (le contenu du fichier est interprété par le serveur).

Avantages

- Le code est rapide à programmer et à exécuter.
- Un même script peut tourner sur différents serveurs WEB (Apache, IIS).
- PHP inclut en standard de nombreuses fonctionnalités.
- PHP est gratuit.
- PHP existe sous de nombreux système d'exploitation (Unix, Linux, Windows & Mac).
- Il est uniquement dédié au développement de page WEB dynamique.
- Plusieurs fournisseurs d'accès ou hébergeurs de sites utilisent PHP.

La notion d'objet

Le traitement d'un objet concerne la valeur de cet objet. Si cette valeur ne peut pas être modifiée, nous parlerons de constante, sinon nous parlerons de variable.

Un objet est parfaitement défini si nous connaissons ses trois caractéristiques :
- Indicateur
- Valeur
- Type

L'indicateur

C'est le nom de l'objet, il est représenté par une suite quelconque de caractères alphanumériques (numériques et alphabétiques sans espace) commençant obligatoirement par une lettre.

Le nom est chou en rapport avec le contenu de l'objet.

Exemple :

Nom	
TOTAL	OK
NOMBRE_DE_PRODUITS	OK
NombreDeProduits	OK
1CUMUL	NON
TOTO	NON
NB LIVRES	NON

Les types

On ne peut pas appliquer de traitement à la valeur d'un objet si on ne connait pas son type. Un type est défini par :

- Un ensemble de constantes.
- Un ensemble d'opérations que l'on peut appliquer.

Il y a trois grands types d'objets :

- Numérique.
- Caractère.
- Booléen.

Ces trois types peuvent être combinés pour donner les types composés :

- Les tableaux.
- Les structures.
- …

PHP n'est pas typé, c'est-à-dire qu'il n'y a pas de définition de variable. Les variables PHP peuvent donc contenir tous types de données :

- Chaînes de caractères - entiers - flottant - …

La seule règle à respecter est que chaque variable doit être précédée de $.
Exemple : $mavariable

On définit les constantes en utilisant la syntaxe suivante :

define ("nomconstante", valeurConstante);

La valeur 0, une chaîne de caractères vide " ", et la chaîne de caractères "0" signifient "FAUX". Tout le reste signifie "VRAI".

Les commentaires

En PHP on utilise : // ou /*

Exemple :

- /* Ceci est un commentaire */
- // Voici un autre commentaire
- /* Et encore
  un autre */

L'affectation

- L'opération consiste à affecter une valeur à un objet (identificateur).
- On représente cette opération avec :
- Un :=
- Ou une flèche orientée à gauche <—

- Exemple :
  IDENTIFICATEUR := VALEUR
  IDENTIFICATEUR <— VALEUR

En PHP $Identificateur = Valeur;

Exercice

Le programme PHP correspondant à l'algorithme suivant qui déclare deux constantes 'MAI,MINI', trois variables entières 'jour, mois, an', deux variables chaînes 'tst'.

- Algo test_affectation C'EST
- CONST MAXI C'EST 100
  MINI C'EST 1
- VAR jour, mois, an : ENTRER
  nom, prenom : CHAINE
  test : BOOLEEN
- DEBUT
  - CO déclaration des variables entières FCO
  - jour := 01; mois :=12; an := 1980
  - CO déclaration des variables chaines FCO
  - nom := "…"
  - prenom := "…"
  - CO déclaration d'un booléen FCO
  - Est := VRAI
- FIN

```
<?php

// déclaration des constantes
Define ("MAXI", 100); ✓
Define ("MINI", 1); ✓

// déclaration des variables entières
$jours = 01; ✓
$mois = 12; ✓
```

$an = 1980; ✓

⚠ **les minuscules et les majuscules.**

// déclaration des chaines de caractères
$nom = "…"; ✓
$prenom = "…"; ✓

// déclaration d'une variable booléenne
~~**$EST= "VRAI";**~~ ✗
$tst =TRUE;

?>

Les instructions

L'instruction de sortie

Cette instruction permet d'afficher des chaînes de caractère à l'écran.

EN ALGO
- **AFFICHER** "chaine de caracteres", variable

EN PHP
- echo
- print
- printf

L'instruction d'entrée

Cette instruction permet de saisir la valeur d'une variable au clavier.

EN ALGO
- **SAISIR** variable

Exercice

Écrire un alto puis le programme PHP qui effectue la saisie de deux entiers I et J au clavier, les additionne dans la variable K, et affiche le résultat à l'écran.

- Algo numero_4 C'EST
- VAR i, j, k : ENTRER
  nom, prenom : CHAINE
  test : BOOLEEN
- DEBUT
  - AFFICHER "Enter I :"; SAISIR i
  - AFFICHER "Enter J :"; SAISIR j
  - K := i+j
  - AFFICHER i, "+", j, "=", k
- FIN

Problème : comment faire pour transférer des variables d'un formulaire HTML vers un programme PHP.

Solution : les données d'un formulaire sont transférées au processus chargé de les traiter, par l'intermédiaire des méthodes HTTP POST ou HTTP GET.

EN PHP

```
<?php
```

- // On récupère les variables
  $i=$_get ['i'] ;
  $j=$_get ['j'];

- // On effectue les calculs
  $k=$i+$j;

- // On affiche le résultat
  echo $i. "+".$j." = ".$k;

  écho "$i+$j = $k";
-

```
?>
```

Exemple de fonctions

- **Longueur de (chaîne).**
  Donne le nombre de caractères situés entre les deux guillemets qui délimitent la chaîne.

  En PHP strlen($chaine).

- **Code (caractère).**
  restitue la valeur de code numérique lié au caractère.

  En PHP ord($caractere).

- **CAR (nombre).**
  restitue le caractère correspondant à la valeur numérique nombre.

  En PHP chr($nombre)

- **Valeur (chaîne).**
  renvoie le nombre contenu dans une chaîne de caractères sous la forme d'une valeur numérique.

  En PHP (integer)$chaine

- **STR (Nombre).**
  renvoie une valeur de type chaîne représentant un nombre.

  En PHP (string)$nombre

Exemple :

```php
<?php

$maChaine="Bonjour";
echo "Taille de la chaine \"Bonjour\" : ".strlen($machaine);
echo "<br />";



$monCar="A";
echo "Code ASCII du 'A' : ".ord($monCar);
echo "<br />";
```

```php
$monEntier=97;
echo "caractere ASCII correspondant au 87 : ".chr($monEntier);
echo "<br />";


$maChaine1="1234";
echo "Conversion de type addition : ".(integer)$maChaine1 +=1 ;
echo "<br />";



$maChaine2=1234;
echo "Conversion de type et concatenation : ".(string)$maChaine2."1" ;
echo "<br />";

?>
```

La structure conditionnelle

- SI condition
  - Alors
- FSI

Lorsque l'évaluation de la condition produit la valeur VRAI l'action est exécutée.

En PHP

- If (condition) {
  - action ;
  - …… ;
- }

Exercice :

Écrire un programme PHP qui demande à l'utilisateur de saisir un entier puis affichent : positif, négatif ou nul en fonction de la valeur de cet entier.

```php
<?php

//On récupère les variables

$x=$_get['x'];



//On effectue les tests

if ($x>0) {
    echo "positif";
}

if ($x<0) {
    echo "Négatif";
}

if ($x==0) {
    echo "nul";
}

?>
```

La structure alternative

Algorithme	En PHP

• SI condition
  • ALORS action1
  • SINON action2
• FSI
	If (condition) {
	  actionsV ;
	}
	else

Lorsque l'évaluation de la condition produit la valeur :

- VRAI : l'acteon1 est exécutée
- FAUX : l'acteon2 est exécutée

Exercice :

Modifier le programme PHP présent en utilisant la structure alternative.

```php
<?php

//On récupère les variables

$x=$_get['x'];



//On effectue les tests

if ($x>0) {
    echo "Positif";

} else {

    if ($x<0) {
        echo "Négatif";

    }else{

        echo "Nul";
    }
}

?>
```

Ou

```php
<?php

//On récupère les variables

$x=$_get['x'];



//On effectue les tests

if ($x==0) {
    echo "Nul";

} else {

    if ($x>0) {
        echo "Positif";

    }else{

        echo "Négatif";
    }
}

?>
```

La structure alternative

Exercice :

Écrire le programme en PHP qui calcule puis affiche la valeur absolue d'un entier entré au clavier.

```php
<?php

//On récupère les variables

$i=$_get['i'];

//On effectue les tests

if ($i>=0) {
    $valabs=$i;
} else {
    $valabs=-$i;
}

echo "|".$i."|=".$valabs;

?>
```

La structure de choix

Lorsque l'on a plusieurs conditions, l'usage du **SI** devient difficile à gérer. C'est pourquoi on utilise la structure de choix.

- **Cas** expire **C'EST**
  - chx1 : instr1
  - ……
  - chxN : instrN
  - **AUTRES :** intrA
- **FCAS**

L'expression est comparée aux différentes constantes et les actions entreprises dépendent de cette valeur. On dispose d'une action par défaut si la variable n'est égale à aucune autres constantes énumérées.

SWITCH

En PHP

```
<?php
switch (expr) {
    case ch1 :
        inst;
        ...........
        breack;

    case ch2 :
        inst;
        ...........
        breack;
.....................
    default:
        instruction;
        ...........;
}


?>
```

Exercice :

Donner le programme en PHP qui permet d'afficher :

• Positif	Si i > 0
• Négatif	Si i < 0
• Nul	Si i = 0
• Ce n'est pas un chiffre	Si i > 9 ou i < -9

En fonction du nombre « i ».

- **Cas** i **C'EST**
  - 0 : Afficher « Nul »
  - 1…9 : Afficher « Positif »
  - -9…-1 : Afficher « Ce n'est pas un chiffre »
  - **AUTRES :** intrA
- **FCAS**

L'instruction « Tant que »

Elle spécifie qu'une instruction, ou un groupe d'instructions doivent être exécutées un certain nombre de fois.

Format général
- **TANT QUE** condition
  - Instruction
  - ……
- **FTQ**

On teste **d'abord** la condition
- Si elle est vraie alors on exécute les instructions.
  - Puis on boucle de nouveau sur le test.
- Si elle est fausse, la boucle se termine.
  - Le programme poursuit son exécution après **FTQ.**

Remarques :

- Le test et effectué avant l'exécution du bloc, il est donc possible de n'avoir aucune exécution du bloc d'instruction.
- Comme il est conseillé de na pas boucler indéfiniment, une instruction au moins doit agir sur l'expression logique pour la rendre vraie.

EN PHP

```
While (cdt) {
        Instr
        ……
}
```

Exercice 1 :

Donner le programme PHP qui permet d'afficher la table caractère ASCII de 33 à 255.

```
echo chr(33);
echo chr(33);
echo chr(n);
```

Ou

En Algo	En PHP

• ALGO numéro_8 C'EST
• VAR i : ENTIER
• DEBUT
  • i :=33
  • TANT QUE i<256
  • AFFICHER i, " ", CAR (i)
  • I := i+1
  • FTQ
• FIN
	<?php
	
	//On récupère les variables
	
	$i=33;
	
	//On effectue les tests
	
	While (i<=255) {
	  echo $i."".chr($i)."
		";
		  $i++;
		
		}
		
		?>

L'instruction Répéter

Format général :

- **Répéter**
  - Instr
  - …
  - …
- **JSQUA** cdt

- **On exécute le bloc d'instructions.**
- **On évalue la condition :**
  - Si la condition est **vraie :** on quitte le **REPETER.**
  - Si la condition est **fausse** on **recommence.**

Avec cette forme répétitive, le bloc d'instructions est **exécuté au moins une fois**.

EN PHP

```
<?php
do {

    Instr ;
    …

}
While (cdt)

?>
```

- La structure est **do … while :** c'est à dire **Faire … TANT QUE.** Alors que la structure algorithmique est répéter … jusqu'à.
- C'est à dire qu'en PHP on exécute l'action tant qu'une condition est vraie alors qu'en algorithme on exécute une action tant que la condition est fausse, c'est à dire jusqu'à ce que la **condition inverse soit vraie**.

Exercice :

Écrire un programme PHP qui permet d'afficher 20 lignes numérotées sur une page web.

```
<?php

// Déclaration d'une vraiable
$i=1;
```

```
    do {

        echo $i.'<br/>';
        $i++;

    }
    while ($i<=20);

?>
```

Remarque : Utilisation de break et du if dans une boucle infinie.

L'instruction pour

Format général :

- **POUR** indice **DE** valDeb **A** valFin **PAS** pas
  - Instruction
  - ……
- **FPOUR**

- Cette structure permet la répétition d'une action un nombre connu de fois.
- La variable `'indice'` est tu type entier.
- La variable initiale et la valeur finale sont des variables de type entier.
- La variable est augmenté de **1** à chaque passage dans la boucle **POUR** lorsque le '**PAS**' n'est pas précisé.

EN PHP

```
for (expression1;expression2;expression3)

{
    instruction;
    ……;
}
```

Expression1 : est une affectation dans laquelle la variable de compteur prend la valeur initiale.

Expression2 : est une expression logique, si le résultat de l'évaluation est **VRAI**, les instructions à l'intérieur de la boucle sont traitées.

Expression3 : est une affectation dans laquelle la valeur de la variable de compteur est augmentée ou diminuée.

Exercice :

Écrire le programme PHP qui permettent d'afficher à l'écran une table de multiplication.

Les fonctions

- Une fonction est un bloc de code qui n'est pas exécuté de manière linéaire dans un script.
- Le code sera exécuté que lors de l'appel de la fonction.
- Écrit une foi, ce code peut être exécuté aussi souvent que nécessaire.
- Cela allège d'autant l'ensemble du code.
- Une fonction retourne la valeur.

La syntaxe

En PHP

```php
<?php

//declaration
function nomfct($param1, …, $paramN)
{
    instr;
    ……;
    ……;
    return valeur;
}

//appel
$valeur=nomfct($param1, …, $paramN);

?>
```

Contraintes :

- Nombre
- Type
- Ordre

Exemple :

```php
<?php

echo"Sans les fonctions …<br /><br />";
$prenom1="Pierre";
$prenom2="Jean";
echo"Bonjour ".$prenom1." ".$prenom2."<br /><br />";

?>

<?php
function bonjour($unprenom1,$unprenom2)
{
    echo"Bonjour ".$unprenom1." ".$unprenom2."<br /><br />";
}
echo"<br /><br />Avec les fonctions …<br /><br />";
$prenom1="Pierre";
$prenom2="Jean";
bonjour($prenom1, $prenom2);
bonjour($prenom2, $prenom1);
bonjour($prenom1, $prenom1);
bonjour($prenom2, $prenom2);
?>
```

Les fonctions paramètres

- Les paramètres ont deux rôles :
  - Transmettre à la fonction, au moment de l'appel, les valeurs nécessaires à son exécution.
  - Au moment du retour, transmettre éventuellement au programme appelant le ou les résultats du traitement effectué.
- On distingue essentiellement deux types de transmission :
  - Le passage de paramètres par valeur, ENTRÉE
    le paramètre **n'est pas modifié** par la procédure.
  - Le passage de paramètres par RÉFÉRENCE, Adresse, Variable, Sortie
    le paramètre **peut être modifié** par la procédure.

Exemple :

```php
<?php

//passage par valeur
function ajoutUn($unEntier)
{
    $unEntier=$unEntier+1;
}
$a=0;
echo"Valeur de \$a avant l'appel
par valeur ";
echo"a la fonction ajoutUn : ".
$a."<br />";


ajoutUn($a);
echo"Valeur de \$a après l'appel
par valeur ";
echo"a la fonction ajoutUn : ".
$a."<br />";

?>
```

```php
<?php

//passage par valeur
function ajoutUnV2(&$unEntier)
{
    $unEntier=$unEntier+1;
}
$a=0;
echo"Valeur de \$a avant l'appel
par valeur ";
echo"a la fonction ajoutUn : ".
$a."<br />";
echo"<br />";

ajoutUnV2($a);
echo"Valeur de \$a après l'appel
par valeur ";
echo"a la fonction ajoutUn : ".
$a."<br />";

?>
```

Exercice :

Écrire en PHP la fonction PERMUT qui effectue la permutation des deux variables entières qui lui sont passés en paramètres, donner aussi le programme PHP testant cette fonction.

```php
<?php
```

```php
//Initialisation
$a=5;
$b=7;
echo"<br />";
echo"Valeur avant les permutations des variables \$a= ".$a." et \$b= ".
$b;


//permutation des variables

function permutation(&$a, &$b)
{
    $c=$a;
    $a=$b;
    $b=$c;
}


//final

permutation($a, $b);

echo"<br />";
echo"Valeur après les permutations des variables \$a= ".$a." et \$b= ".
$b;

?>
```

Les tableaux

Définition

Dans beaucoup de problèmes pratiques, le fait de devoir définir une variable, avec un nom unique pour chaque valeur manipulée dans le programme pose des problèmes de faisabilité. C'est pour cette raison que les langages de programmation offrent la possibilité de définir des tableaux.

Les tableaux (**array**) représentent la structure de données la plus importante du langage PHP, c'est **une structure de données linéaire** qui permet de stocker des données de **type différent**.

Les valeurs sont appelées **éléments** du tableau et sont repérées par **un indice** indiquant la position relative de la donnée par rapport au début du tableau.
Les index de tableau en PHP commencent à **0**.

Déclaration d'un tableau vide

- $monTableau = **array();**

Déclaration d'un tableau avec des données

- $prenoms = **array**("Pierre","Paul","Jacques");
- $entiers = **array**(45,32,25);

Déclaration d'un tableau commençant à 1

- $entiers1 = **array**(1 => 4,12,17);

Utilisation

L'accès à un élément du tableau se fait en spécifiant le nom du tableau, suivi de l'indice placé entre crochets.

Exemple :

```
$entiers[0]=9;
$entiers[0]=$entiers[0]+1;
echo $entiers[0];

$prenoms[4]="michel";
echo $prenoms[4];
```

fct()	Fonction
care ‘ ‘	Caractère
chaines” “	
prog{ }	
Tableau [ ]	

Les tableaux Count

La fonction **Count** permet de compter tous les éléments d'un tableau, elle renvoie 0 si le tableau est vide ou non affecté.

Exemple :

```php
<?php

$villes=array();
echo "Count de villes : ".count($villes);
echo "<br/>";

$villes[0]="Brest";
$villes[1]="Quimper";
$villes[2]="Rennes";
echo "Count de villes : ".count($villes);

?>
```

Les tableaux initialisation

En utilisant la fonction **rand**(min,max) et la boucle **for** créer un tableau contenant 30 nombres entre 1 et 100.

Code :

```php
<?php

$nobres=array();

for($i=0;$i<30;$i++){
    $nombres[$i]=rand(1,100);
}
print_r($nombres);

?>
```

Les tableaux affichage

On parcourt les différentes case du tableau en faisant varier l'indice et on affiche son contenu au fur et à mesure en utilisant la boucle **for** et la fonction **Count**.

Code :

```php
<?php

$nombres=array();

for($i=0;$i<30;$i++){
    $nombres[$i]=rand(1,100);
}
print_r($nombres);
echo "<br/>";
echo "<br/>";

for($i=0;$i<count($nombres);$i++){
    $nombres[$i]=rand(1,100);
    echo "nombre ".$i." : ";
    echo $nombres[$i]."<br/>";
}

?>
```

La fonction **print_r** affiche des informations lisibles pour une variable (utiliser en debug). L'instruction foreach c'est un moyen simple de passer en revue un tableau, elle fonctionne uniquement sur les tableaux. (depuis PHP5 foreach fonctionne aussi avec les objets).

Code :

```php
<?php

$nombres=array();

for($i=0;$i<30;$i++){
    $nombres[$i]=rand(1,100);
}

print_r($nombres);
echo "<br/>";
echo "<br/>";

echo "<br/>";
echo "<br/>";
```

```
foreach ($nombres as $valeur){
     echo "Nombre ".$valeur."<br/>";
 }

?>
```

Les tableaux associatif

Chaque éléments d'un tableau peut aussi être identifié par une étiquette, qui est une chaîne de caractères ou une variable de type **string**, nommé clef.

Déclaration d'un tableau associatif :

```
$monTableauAssoc=array("numero"=>1234,
                       "nom"=>"Durand",
                       “prenom"=>"Jean");
```

Utilisation d'un tableau associatif :

```
$monTableauAssoc=array("numero"=>1234,
                       "nom"=>"Durand",
                       "prenom"=>"Jean");
 echo $monTableauAssoc['nom'];
 echo $monTableauAssoc[‘prenom'];
```

Exercice

Ecrire un programme PHP qui définit un tableau associatif contenant des informations sur un employé : (Numéro, Nom, Prénom, Adresse, CP, Ville, Salaire) puis affiche le type et la valeur pour chaque élément du tableau associatif $unEmploye et affiche l'état suivant :

NOM Prénom
Adresse
Code Postal Ville

```
$unEmploye=array("numero"=>1234,
                 "nom"=>"Durand",
                 "prenom"=>"Jean",
                 "adresse"=>"32 Rue de Quimper",
                 "CP"=>"29200",
```

```
                    "ville"=>"Brest",
                     "Salaire"=>"1599 €"
 );

echo "<br/>";
echo "<br/>";


echo $unEmploye['numero'];
echo "<br/>";
echo "<fieldset>";
echo $unEmploye['nom'];
echo " ";
echo $unEmploye['prenom'];
echo "<br/>";
echo $unEmploye['adresse'];
echo "<br/>";
echo $unEmploye['CP'];
echo " ";
echo $unEmploye['ville'];
echo "</fieldset>";
echo $unEmploye['Salaire'];

echo "<br/>";
echo "<br/>";


?>
```