Les structures itératives en PHP

1. Objectifs

- Connaitre les structures itératives en PHP

2. Introduction

- Une structure itérative (ou boucle) est un bloc d'instructions dont le but est répéter une opération tant qu'une ou plusieurs conditions sont vérifiées. PHP en distingue 4 :

  `while` `do while` `for` `foreach`

- Les boucles permettent de répéter des opérations élémentaires un grand nombre de fois sans avoir à réécrire le même code. Selon l'instruction de boucle utilisée, le nombre d'itérations peut être défini à l'avance ou être déterminé par une condition particulière.

3. La boucle `for`

- La boucle **for** permet d'exécuter plusieurs fois la même instruction ou le même bloc sans avoir à réécrire les mêmes instructions.
- > ***La boucle for en PHP est utilisée pour exécuter une séquence d'instructions un certain nombre de fois. Elle est principalement utile lorsque vous connaissez le nombre d'itérations à l'avance.***

4. Syntaxe

```
for (initialisation; condition; itération) {
    // Instructions à exécuter
}
```

- ***L'initialisation*** est exécutée une seule fois au début de la boucle. C'est là que vous définissez et initialisez la variable de contrôle.
- ***La condition*** est évaluée avant chaque itération. Si la condition est vraie (true), la boucle continue à s'exécuter. Si la condition devient fausse (false), la boucle s'arrête.
- ***L'itération*** est exécutée à la fin de chaque itération de la boucle. Vous pouvez y ajuster la variable de contrôle.

5. Exercices d'application

apcpedagogie.com

1. **Exercice 01**

   - **Enoncé**
     - Créer un script PHP qui affiche six niveaux de titre utilisant les balises <h1> à <h6> et une couleur bleue en deux lignes de code seulement.
     - Le résultat de la boucle est illustré à la figure suivante:

   Solution

```
<?php
for($i=1;$i<7;$i++)
{
    echo "<font color='blue'><h$i> $i:Titre de niveau $i</font></h$i>";
}
?>
```

2. **Exercice 02**

   - **Enoncé**
     - Créez une page web qui affiche les nombres **pairs** et **impairs** jusqu'à une **constante spécifiée**. Voici les instructions :
       - Créez un fichier HTML avec une structure de base, y compris une balise **<h1>** pour le titre.
       - Utilisez une feuille de style CSS pour ajouter une mise en forme à votre page, par exemple en utilisant des couleurs différentes pour les nombres pairs et impairs.
       - Dans votre code PHP, définissez une constante pour le nombre maximal jusqu'auquel vous souhaitez afficher les nombres pairs et impairs.
       - Utilisez une boucle for pour parcourir les nombres de 1 jusqu'à la constante.
       - À chaque itération, vérifiez si le nombre est pair ou impair.
       - Utilisez du HTML pour afficher le nombre, et indiquez s'il est pair ou impair en utilisant des classes CSS.
       - Testez votre page en modifiant la valeur de la constante pour afficher les nombres jusqu'à la limite souhaitée.
       - N'oubliez pas d'inclure des commentaires dans votre code pour expliquer chaque étape. Ce projet vous permettra de pratiquer

apcpedagogie.com

l'utilisation de PHP en combinaison avec HTML et CSS pour afficher des nombres pairs et impairs de manière élégante.

Solution

- **CSS (styles.css) :**

```
body {
    font-family: Arial, sans-serif;
    text-align: center;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
}

h1 {
    color: #333;
}

#numbers {
    margin: 20px;
    padding: 10px;
    border: 2px solid #333;
    background-color: #fff;
    display: inline-block;
    border-radius: 5px;
}

.even {
    color: green;
}

.odd {
    color: red;
}
```

- **PHP (numbers.php) :**

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="styles.css">
    <title>Affichage de nombres pairs et impairs</title>
```

apcpedagogie.com

```
</head>
<body>
    <h1>Nombres pairs et impairs jusqu'à une constante</h1>
    <div id="numbers">
        <?php
        define('CONSTANT', 10); // Définir la constante jusqu'à
laquelle afficher

        for ($i = 1; $i <= CONSTANT; $i++) {
            if ($i % 2 == 0) {
                echo "<p class='even'>$i (Pair)</p>";
            } else {
                echo "<p class='odd'>$i (Impair)</p>";
            }
        }
        ?>
    </div>
</body>
</html>
```

3. **Exercice 03**

   - **Enoncé**
     - Écrivez un script PHP qui génère une table de multiplication de 1 à 10. Voici les étapes à suivre :
       - Utilisez une boucle for pour générer les lignes de la table (nombres de 1 à 10).
       - À l'intérieur de la première boucle for, utilisez une autre boucle for pour générer les cellules de la table.
       - Dans chaque cellule, affichez le résultat de la multiplication entre le numéro de ligne et le numéro de colonne.
       - Utilisez une structure HTML (table) pour afficher la table de multiplication.
       - Testez votre script pour vérifier que la table est correctement générée pour les nombres de 1 à 10.
     - Supposons que vous ayez généré avec succès une table de multiplication de 1 à 10 en utilisant un script PHP, comme indiqué dans l'exercice précédent. Vous avez une structure HTML qui représente la table, mais elle n'a pas encore de styles CSS.

apcpedagogie.com

- Votre tâche consiste à ajouter des styles CSS à la table de multiplication pour la rendre plus attrayante et lisible. Voici quelques points à considérer :
  - Appliquez des bordures et des marges pour délimiter les cellules de la table.
  - Donnez un style différent aux lignes paires et aux lignes impaires pour améliorer la lisibilité.
  - Ajoutez un fond coloré ou une couleur de texte différente pour les en-têtes de colonne (1, 2, 3, ..., 10) et les en-têtes de ligne (1, 2, 3, ..., 10).
  - Assurez-vous que le texte est centré dans les cellules de la table.
  - **Question bonus**: Comment feriez-vous pour que la table soit responsive, c'est-à-dire qu'elle s'adapte aux différentes tailles d'écran ?
  - Cette question vous permet de pratiquer la création de règles CSS pour améliorer l'apparence de la table de multiplication et d'explorer des techniques de mise en page CSS pour la rendre adaptative aux différentes tailles d'écran.

Solution

- **CSS (styles.css) :**

```
table {
    border-collapse: collapse;
    width: 80%;
    margin: 0 auto;
}

th, td {
    border: 1px solid #000;
    padding: 8px;
    text-align: center;
}

th {
    background-color: #f2f2f2;
}

tr:nth-child(even) {
    background-color: #f2f2f2;
}
```

apcpedagogie.com

```
tr:nth-child(odd) {
    background-color: #fff;
}
```

- **CSS (styles.css) :**

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="style.css">
    <title>Table de multiplication</title>
</head>
<body>
    <h1>Table de multiplication de 1 à 10</h1>
    <table>
        <tr>
            <th>x</th>
            <?php
            for ($i = 1; $i <= 10; $i++) {
                echo "<th>$i</th>";
            }
            ?>
        </tr>
        <?php
        for ($i = 1; $i <= 10; $i++) {
            echo "<tr><th>$i</th>";
            for ($j = 1; $j <= 10; $j++) {
                echo "<td>" . ($i * $j) . "</td>";
            }
            echo "</tr>";
        }
        ?>
    </table>
</body>
</html>
```

La boucle `While`

1. **Présentation**

- En anglais "**`while`**" signifie "**Tant que**". Pour créer une boucle , il faut donc utiliser ce mot clé suivi d'une indication qui dit quand la boucle s'arrête.
- Lorsque l'on souhaite effectuer une même série d'instructions tant qu'une condition donnée reste vérifiée, on utilise une boucle "**while**".
- Cette boucle "**while**" est utilisée lorsqu'elle correspond au cas où l'on ne peut déterminer à l'avance le nombre d'itérations nécessaires à la résolution de notre problème.
- Le but de la boucle "**while**" est de répéter certaines instructions tant qu'une condition est respectée.
- L'avantage de "**while**" est donc qu'il ne faut pas connaitre à l'avance le nombre de fois où on devra répéter notre boucle.

2. **Syntaxe**

```
while condition booléenne:
    Instruction 1 # Tant que la condition booléenne est vérifiée
 Instruction 2  # Faire les différentes instructions qui suivent
.......
 Instruction n
```

3. **Exemple:**

- L'exemple ci-dessous effectue une suite de tirages de nombres aléatoires compris entre 1 et 100 grâce à la fonction **rand()**, avec comme condition supplémentaire que le nombre tiré soit un multiple de 7.
- Le script affiche les nombres tirés jusqu'à trouver un multiple de 7.

```
<!DOCTYPE html >
<html lang="fr">
<head>
<title>Les structures de contrôle en PHP</title>
<meta ="UTF-8">
</head>
<body>
<?php
$n=1;
while($n%7!=0 )
{
$n = rand(1,100);
```

apcpedagogie.com

```
echo $n,"  /";
}

?>
</body>
</html>
```

- ## La boucle `do.. While`

1. **Définition et utilisation**
   - La structure **`do`** permet d'exécuter une instruction ou un bloc d'instructions et de répéter cette action tant qu'une expression logique controlée par **while** est vraie.
   - Les instructions **`do..while`** sont similaires aux instructions **`while`**, si ce n'est que la condition qui est vérifiée à la fin de chaque itération, et non au début.
   - Cela signifie que la boucle s'exécute toujours au moins une fois (l'expression n'est testée qu'à la fin de l'itération), ce qui n'est pas le cas lorsque vous utilisez une boucle **`while`** (l'expression est vérifiée au début de chaque itération).
2. **Syntaxe**

```
do
{
instructions
} while (condition(s))
```

Paramètres

- **instruction**
- Une instruction exécutée au moins une fois et ré-exécutée chaque fois que la condition de test est évaluée à true. On peut exécuter plusieurs instructions au sein d'une boucle grâce à l'instruction block ({ ... }) qui permet de grouper différentes instructions en une seule.
- **condition**
- Une expression évaluée après chaque passage dans la boucle.

3. **Exemple :**

apcpedagogie.com

```
<!DOCTYPE html >
<html lang="fr">
<head>
<title>Les structures de contrôle en PHP</title>
<meta ="UTF-8">
</head>
<body>
<?php
$i = 0;
do {
    echo $i ." ";
    $i++;
} while ($i < 10);

?>
</body>
</html>
```

4. Dans l'exemple suivant, les instructions de la boucle **do...while** continuent à s'exécuter tant que la variable i est inférieure à 10.

- ## La boucle `foreach`

1. **Présentation**

    - La boucle **`foreach`** est particulièrement efficace pour lister les tableaux associatifs dont il n'est nécessaire de connaître ni le nombre d'éléments ni les clés.
    - Le but de la boucle **`foreach`** est d'effectue une itération spécifique sur les éléments du tableau(**Array**).
    - Sa syntaxe est variable selon que vous souhaitez récupérer seulement les valeurs ou les valeurs et les clés (ou les indices).

2. **Syntaxe**

```
foreach($tableau as $valeur)
{
//bloc utilisant la valeur de l'élément courant
}
```

apcpedagogie.com

3. La variable `$valeur` contient successivement chacune des valeurs du tableau. Il importe cependant de ne pas utiliser un nom de variable existant, faute de quoi sa valeur est écrasée.
4. Les variables utilisées dans une boucle `foreach` ne sont pas locales à la boucle et gardent donc la valeur du dernier élément lu dans tout le script.
5. Pour plus d'information consultez l'article parcourir les tableaux en PHP

Les instructions d'arrêt et de continuité

- **PHP** introduit également deux instructions particulières des boucles. Il s'agit de `continue` et `break`.
- La première permet de forcer le passage à l'itération suivante en sautant tout ou partie du bloc d'instructions à exécuter.
- La seconde, quant à elle, permet de forcer à quitter une structure conditionnelle telle que **`for()`**, **`while()`**, **`foreach()`** ou **`switch()`**.
- Ces deux instructions peuvent prendre un paramètre optionnel de type entier permettant de connaître le nombre de structures emboîtées qui ont été interrompues.
- Illustrons brièvement ces deux concepts à l'aide d'exemples simples.

```
Exemple d'utilisation de continue
<?php
  for($i=0; $i<=10; $i++)
  {
    // On n'affiche pas les 5 premiers nombres
    if($i<=5)
    {
      continue;
    }
    // Affichage des nombres de 6 à 10
    echo $i ,'<br/>';
  }
?>
```

Applications

1. **App01**
    - Effectuer une suite de tirages de **nombres aléatoires jusqu'à** obtenir une **suite composée** d'un nombre pair suivi de deux nombres impairs.

apcpedagogie.com

- Arrêter les tirages et afficher le nombre de coups réalisés.
- Exemple de résultat obtenu :

  194,285,494 435,759,162 237,292,768 **366,533,397** Résultat obtenu en 4 coups

  Solution

```
<?php
$compteur=0;
do
{
  $x=rand(100,1000);
  $y=rand(100,1000);
  $z=rand(100,1000);

      if($x%2==1 OR $y%2==0 OR $z%2==0){
          echo $x, ", ",$y, ", ",$z,"<br />";
        }else{
            echo "<b>",$x, ", ",$y, ", ",$z,"</b><br>";
        }
 $compteur++;
}while($x%2==1 OR $y%2==0 OR $z%2==0);
echo "Résultat obtenu en $compteur coups";
?>
```

2. **App02**

- Choisir un nombre de **trois chiffres**.
- Effectuer ensuite des tirages aléatoires et compter le nombre de tirages nécessaire pour obtenir le nombre initial. Arrêter les tirages et afficher le nombre de coups réalisés.
- Réaliser ce script d'abord avec l'instruction `while` puis avec l'instruction `for`.
- **l'instruction** `while`

  Solution

```
<?php
//Nombre à trouver
         $nb=789;
         //compteur
         $coup=0;
         $x=0;
         //boucle de tirage
```

apcpedagogie.com

```
          while($x!=$nb)
          {
             $x=rand(100,1000);
             $coup++;
             //echo $x,"<br />";//pour afficher tous les tirages
          }
          echo "<h1>$nb trouvé en $coup coups </h1>";
?>
```

- **l'instruction** `for`

  Solution

```
<?php
  //Nombre à trouver
  $nb=789;
  $x=0;
 //boucle de tirage
 for($coup=1;$x!=$nb;$coup++)
 {
   $x=rand(100,1000);
   //echo $x,"<br />";//pour afficher tous les tirages
 }
          echo "<h1>$nb trouvé en $coup coups </h1>";
?>
```

3. **App03**

- Utiliser une boucle `while` pour déterminer le premier entier obtenu par tirage aléatoire qui soit un multiple d'un nombre donné.
- Écrire la variante utilisant la boucle **`do…while`**.
- **l'instruction** `while`

  Solution

```
<?php
$nb=57;
$compteur=0;
$x=rand(10,1000);
 //Boucle while
while($x%$nb!=0)
  {
```

apcpedagogie.com

```
 $x=rand(10,1000);
  $compteur++;
 }
  echo "<h1>$x est multiple de $nb: Résultat obtenu en $compteur coups</h1>";
 ?>
```

- **l'instruction `do…while`**

  Solution

```
<?php
 $nb=57;
 $compteur=0;
 do
 {
 $x=rand(0,1000);
 $compteur++;
  }
  while($x%$nb!=0);
 echo "<h1>$x est multiple de $nb: Résultat obtenu en $compteur coups</h1>";
?>
```

- ***Références***:
  - *https://www.php.net*
  - *Livre PHP7 Cours et exercices Jean Engels 2e édition*
  - *Technologies de l'Information et de la Communication 4 année secondaire*

Sommaire du cours PHP

Source : https://apcpedagogie.com/les-structures-iteratives-en-php · téléchargé le 08/11/2026 · © APC Pédagogie · apcpedagogie.com