PHP : structures de contrôle

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

elmouelhi.achref@gmail.com

Plan

PHP

Exécuter si une condition est vraie

```
if (condition) {
    ...
}
```

PHP

Exécuter si une condition est vraie

```
if (condition) {
    ...
}
```


Pour les conditions, on utilise des opérateurs de comparaison

PHP

Opérateurs de comparaison

- == : pour tester l'égalité des valeurs
- != ou <> : pour tester l'inégalité des valeurs
- === : pour tester l'égalité des valeurs et des types
- !== : pour tester l'inégalité des valeurs ou des types
- > : supérieur à
- < : inférieur à
- >= : supérieur ou égal à
- <= : inférieur ou égal à

PHP

Exemple

```
$x = 3;
if ($x >= 0) {
    echo $x . " est positif";
}
```

PHP

Remarques

- En **PHP**, les valeurs suivantes sont considérées comme `falsy` lorsqu'elles sont évaluées dans un contexte booléen :
  - `False` : la valeur booléenne.
  - `NULL` : la valeur nulle.
  - `0` : la valeur entière zéro.
  - `0.0` : la valeur flottante zéro.
  - `''` : la chaîne de caractères vide.
  - `'0'` : une chaîne contenant seulement le caractère zéro.
  - `[]` ou `array()` : le tableau vide.
- Toutes les autres valeurs en **PHP** sont considérées comme `truthy` et seront évaluées comme `true` dans des contextes booléens.

PHP

Voici un exemple de code illustrant cela

```php
if (0) {
    echo "Ceci ne sera pas affiché.";
}

if ("") {
    echo "Ceci ne sera pas affiché.";
}
```

PHP

Exécuter un premier bloc si une condition est vraie, un deuxième sinon (le bloc `else`)

```
if (condition1) {
   ...
} else {
   ...
}
```

PHP

Exécuter un premier bloc si une condition est vraie, un deuxième sinon (le bloc `else`)

```
if (condition1) {
   ...
} else {
   ...
}
```


Exemple

```
$x = 3;
if ($x > 0) {
  echo $x . " est strictement positif";
} else {
  echo $x . " est négatif ou nul";
}
```

PHP

On peut enchaîner les conditions avec `else if` (et avoir un troisième bloc voire ... un nième)

```
if (condition1) {
    ...
} else if (condition2) {
    ...
}
...
else {
    ...
}
```

PHP

On peut enchaîner les conditions avec `else if` (et avoir un troisième bloc voire ... un nième)

```
if (condition1) {
    ...
} else if (condition2) {
    ...
}
...
else {
    ...
}
```

On peut écrire `elseif` ou `else if`.

PHP

Exemple

```
$x = -3;
if ($x > 0) {
  echo $x . " est strictement positif";
} else if ($x < 0) {
  echo $x . " est strictement négatif";
} else {
  echo $x . " est nul";
}
```

PHP

Exercice

Écrire un script **PHP** qui permet de déterminer si une chaîne de caractères `$ma_chaine` est vide ou si elle contient un nombre pair ou impair de caractères.

PHP

Correction

```
<?php
    $ma_chaine = "";
    $longueur = strlen($ma_chaine);
    if ($longueur == 0) {
        echo "'$ma_chaine' est vide";
    } else if (0 == $longueur % 2) {
        echo "'$ma_chaine' a un nombre pair de caractères";
    } else {
        echo "'$ma_chaine' a un nombre impair de caractères";
    }
?>
```

PHP

Étant données les deux chaînes de caractères suivantes

```
$ma_chaine = "Hello les holoulos";
$motif = "lo";
```

PHP

Étant données les deux chaînes de caractères suivantes

```
$ma_chaine = "Hello les holoulos";
$motif = "lo";
```


Exercice

Écrire un script **PHP** qui permet de vérifier si une chaîne de caractères `$ma_chaine` contient deux occurrences de `$motif`.

PHP

Correction

```
<?php
  if (strrpos($ma_chaine, $motif) != strpos($ma_chaine, $motif)) {
    echo "'$ma_chaine' contient au moins deux occurences de '$motif'";
  } else {
    echo "'$ma_chaine' ne pas contient deux occurences de '$motif'";
  }
?>
```

PHP

Opérateurs logiques

- `&&` ou `and` : et
- `||` ou `or` : ou
- `!` : non
- `xor` : ou exclusif

PHP

Opérateurs logiques

- `&&` ou `and` : et
- `||` ou `or` : ou
- `!` : non
- `xor` : ou exclusif

Tester plusieurs conditions (en utilisant des opérateurs logiques)

```
if (condition1 && !condition2 || condition3) {
    ...
}
[else ...]
```

PHP

Exercice 1

Écrire un code **PHP** qui

- demande à l'utilisateur de saisir une année (un entier),
- affiche si l'année saisie est bissextile (voir `https://fr.wikipedia.org/wiki/Ann%C3%A9e_bissextile`).

PHP

Exercice 2

Écrire un code **PHP** qui

- demande à l'utilisateur de saisir deux entiers `a` et `b` différents de zéro,
- affiche le signe du résultat de la multiplication sans calculer le produit.

PHP

Exercice 3

Écrire un code **PHP** qui

- demande à l'utilisateur de saisir deux entiers `a` et `b`,
- détermine et affiche si le résultat de l'addition (sans calculer la somme) est pair ou impair.

PHP

Structure conditionnelle `switch` : syntaxe

```
switch ($nom_variable) {
  case constante-1:
    groupe-instructions-1;
    break;
  case constante-2:
    groupe-instructions-2;
    break;
  ...
  case constante-N:
    groupe-instructions-N;
    break;
  default:
    groupe-instructions-par-défaut;
}
```

PHP

Remarques

- Le `switch` permet **seulement** de tester l'égalité
- Le `break` permet de quitter le `switch` une fois un bloc `case` exécuté
- Il est possible de regrouper plusieurs `case`
- Le bloc `default` est facultatif, il sera exécuter si la valeur de la variable ne correspond à aucune constante de `case`

PHP

Structure conditionnelle avec `switch`

```
$x = 5;
switch ($x) {
    case 1:
      echo("un");
      break;
    case 2:
      echo("deux");
      break;
    case 3:
      echo("trois");
      break;
    default:
      echo("autre");
}
```

PHP

Un multi-case pour un seul traitement

```
$x = 5;
switch ($x) {
  case 1:
  case 2:
    echo("un ou deux");
    break;
  case 3:
    echo("trois");
    break;
  case 4:
  case 5:
    echo("quatre ou cinq");
    break;
  default:
    echo("autre");
}
```

PHP

Si on supprime un `break`

```
$x = "2";
switch ($x) {
    case "1":
      echo("un");
      break;
    case "2":
      echo("deux");
    case "3":
      echo("trois");
      break;
    default:
      echo("autre");
}
/* affiche deuxtrois */
```

PHP

Exercice

Écrire un script **PHP** qui permet d'afficher le nombre de jours d'un mois selon son indice.

Correction

```php
$indice_mois = 1;
switch ($indice_mois) {
    case '1':
    case '3':
    case '5':
    case '7':
    case '8':
    case '10':
    case '12':
        echo 31;
        break;
    case '4':
    case '6':
    case '9':
    case '11':
        echo 30;
        break;
    case '2':
        echo '28 ou 29';
        break;
    default:
        echo "erreur";
        break;
}
```

PHP

match

- Introduit dans **PHP 8**
- Fonctionne comme `switch` mais
  - Pas besoin de `break` ni de `case`
  - $\Rightarrow$ sépare la partie valeur à comparer de la partie traitement
  - Une seule instruction autorisée après $\Rightarrow$
  - Retourne une valeur
  - Effectue des comparaisons strictes

PHP

Exemple

```php
$x = 5;
$resultat = match ($x) {
    1 => "un",
    2 => "deux",
    3, 4 => "trois ou quatre",
    default => "autre",
};

echo $resultat;
// affiche 5
```

PHP

Simplifions l'écriture avec l'expression ternaire (Elvis operator)

```php
$x = 2;
$type = ($x % 2 == 0) ? "pair" : "impair";
echo($type);
/* affiche pair */
```

PHP

Simplifions l'écriture avec l'expression ternaire (Elvis operator)

```
$x = 2;
$type = ($x % 2 == 0) ? "pair" : "impair";
echo($type);
/* affiche pair */
```

La version simplifiée

```
$x = 2;
$type = ($x % 2 != 0) ?: "pair";
echo($type);
/* affiche pair */
```

PHP

Pour tester si une variable existe ou si elle a une valeur différente de `null`, on peut utiliser Null coalescing (`??`)

```php
$result = $y ?? 5;
echo($result);
/* affiche 5 car $y n'existe pas */

$x = null;
$result = $x ?? 5;
echo($result);
/* affiche 5 car $x a la valeur null */
```

PHP

Boucle `while` : à chaque itération on teste si la condition est vraie avant d'accéder aux traitements

```
while (condition[s]) {
   ...
}
```

PHP

Boucle `while` : à chaque itération on teste si la condition est vraie avant d'accéder aux traitements

```
while (condition[s]) {
   ...
}
```


Attention aux boucles infinies, vérifier que la condition d'arrêt sera bien atteinte après un certain nombre d'itérations.

PHP

Exemple

```php
$i = 0;
while ($i < 5) {
  echo $i, "<br>";
  $i++;
}
```

PHP

Exemple

```
$i = 0;
while ($i < 5) {
  echo $i, "<br>";
  $i++;
}
```


Le résultat est

```
0
1
2
3
4
```

PHP

Exercice 1

Écrire un script **PHP** qui permet d'afficher les nombres pairs inférieurs à 10.

PHP

Étant donnée la chaîne de caractères suivante

```
$ma_chaine = "hello les holoulos";
```

PHP

Étant donnée la chaîne de caractères suivante

```
$ma_chaine = "hello les holoulos";
```

Exercice 2

Écrire un script **PHP** qui permet de remplacer chaque caractère d'indice impair de la chaine de caractère `$ma_chaine` par son équivalent en majuscule.

PHP

Correction

```php
<?php
    $i = 0;
    while ($i < strlen($ma_chaine)) {
        $ma_chaine[$i] = strtoupper($ma_chaine[$i]);
        $i += 2;
    }
    echo $ma_chaine;
?>
```

PHP

La Boucle `do` ... `while` exécute le bloc au moins une fois ensuite elle vérifie la condition

```
do {
   ...
}
while (condition[s]);
```

PHP

La Boucle `do` ... `while` exécute le bloc au moins une fois ensuite elle vérifie la condition

```
do {
   ...
}
while (condition[s]);
```


Attention aux boucles infinies, vérifier que la condition d'arrêt sera bien atteinte après un certain nombre d'itérations.

PHP

Exemple

```
$i = 0;
do {
  echo $i, "<br>";
  $i++;
} while ($i < 5);
```

PHP

Exemple

```
$i = 0;
do {
  echo $i, "<br>";
  $i++;
} while ($i < 5);
```


Le résultat est

```
0
1
2
3
4
```

PHP

Exercice 1

Écrire un script **PHP** qui permet de demander à l'utilisateur de saisir un nombre compris entre 1 et 10. Nous devons nous assurer que l'utilisateur fournit une entrée valide avant de continuer.

PHP

Correction

```php
<?php
do {
    $nombre = (int) readline("Entrez un nombre entre 1 et 10 : ");
} while ($nombre < 1 || $nombre > 10);

echo "Vous avez saisi un nombre valide : $nombre";
?>
```

PHP

Étant données les deux chaînes de caractères suivantes

```
$chaine = "bonjour tout le monde";
$motif = "o";
```

PHP

Étant données les deux chaînes de caractères suivantes

```
$chaine = "bonjour tout le monde";
$motif = "o";
```


Exercice 2

Écrire un script **PHP** qui permet d'afficher les trois premières positions de la lettre `'o'` dans `$chaine`.

PHP

Étant donnée la chaîne de caractères suivante

```
$ma_chaine = "hello les holoulos";
```

PHP

Étant donnée la chaîne de caractères suivante

```
$ma_chaine = "hello les holoulos";
```

Exercice 3

Écrire un script **PHP** qui permet de compter le nombre de voyelles dans la chaine de caractère `$ma_chaine`.

PHP

Correction

```
<?php
    $ma_chaine = "hello les holoulos";
    $voyelles = "aeouiy";
    $nbr_voyelles = 0;
    $i = 0;
    do {
        if (strpos($voyelles, strtolower($ma_chaine[
           $i])) !== false) {
            $nbr_voyelles++;
        }
        $i++;
    } while ($i < strlen($ma_chaine));
    echo $nbr_voyelles;
?>
```

PHP

Boucle `for`

```
for (initialisation; condition[s]; incrémentation) {
   ...
}
```

PHP

Boucle `for`

```
for (initialisation; condition[s]; incrémentation) {
   ...
}
```


Attention aux boucles infinies si vous modifiez la valeur du compteur à l'intérieur de la boucle.

PHP

Exemple

```
for ($i = 0; $i < 5; $i++) {
  echo $i, "<br>";
}
```

PHP

Exemple

```
for ($i = 0; $i < 5; $i++) {
  echo $i, "<br>";
}
```


Le résultat est

```
0
1
2
3
4
```

Exercice

Écrire un code **PHP** qui permet d'afficher les nombres pairs compris entre 0 et 10.

Exercice

Écrire un code **PHP** qui permet d'afficher les nombres pairs compris entre 0 et 10.

Première solution

```
for ($i = 0; $i < 10; $i++) {
  if ($i % 2 == 0) {
    echo $i, "<br>";
  }
}
```

Exercice

Écrire un code **PHP** qui permet d'afficher les nombres pairs compris entre 0 et 10.

Première solution

```
for ($i = 0; $i < 10; $i++) {
  if ($i % 2 == 0) {
    echo $i, "<br>";
  }
}
```

Deuxième solution

```
for ($i = 0; $i < 10; $i += 2) {
  echo $i, "<br>";
}
```

PHP

Étant donnée la chaîne de caractères suivante

```
$ma_chaine = "hello les holoulos. C'est toujours le
   confinement.";
```

PHP

Étant donnée la chaîne de caractères suivante

```
$ma_chaine = "hello les holoulos. C'est toujours le
   confinement.";
```


Exercice

Écrire un script **PHP** qui permet de compter le nombre de phrases et celui de mots de la chaine de caractère `$ma_chaine`.

PHP

Correction

```
<?php
    $ma_chaine = "hello les holoulos. C'est toujours le confinement.";
    $nbr_phrases = 0;
    $nbr_mots = strlen($ma_chaine) > 0 ? 1 : 0;
    for ($i = 0; $i < strlen($ma_chaine); $i++) {
        if ($ma_chaine[$i] == " ") {
            $nbr_mots++;
        } elseif ($ma_chaine[$i] == ".") {
            $nbr_phrases++;
        }
    }
    echo "#phrases = $nbr_phrases <br> #mots = $nbr_mots";
?>
```

Étant donnée la variable suivante

```
$nombre = 5;
```

PHP

Étant donnée la variable suivante

```
$nombre = 5;
```

Exercice

Écrire un script **PHP** qui permet d'afficher les tables d'addition, soustraction, multiplication et division de `$nombre`.

PHP

Correction

```
<body>
    <div class=container>
        <?php
        for ($j = 0; $j < strlen($op); $j++) {
            echo "<div>";
            for ($i = 1; $i <= 10; $i++) {
                $operation = $nombre . " " . $op[$j] . " " . $i;
                echo "$operation = ";
                eval("echo $operation .'<br>';");
            }
            echo "</div>";
        }
        ?>
    </div>
</body>
```

PHP

Remarques

Dans ces structures itératives et conditionnelles, on peut utiliser :

- `break` : pour quitter la boucle
- `continue` : pour ignorer l'itération courante
- `goto` : pour renvoyer vers un label.

PHP

Exemple avec `break`

```
$j = 5;
do {
    echo $j . "<br>";
    if ($j == 3) {
        break;
    }
    $j--;
} while ($j > 0);
```

PHP

Exemple avec `break`

```
$j = 5;
do {
    echo $j . "<br>";
    if ($j == 3) {
        break;
    }
    $j--;
} while ($j > 0);
```


Résultat

```
5
4
3
```

PHP

Le `break` avec un nombre permet de sortir de plusieurs niveaux de structure de contrôle imbriqués, pas seulement du `switch`

```php
$j = 1;
switch ($j) {
    case 1:
        while (true) {
            echo "Dans la boucle et le switch\n";
            break 2; // quitte le while ET le switch
        }
        echo "Jamais exécuté\n";
        break;
    default:
        echo "Autre cas\n";
}
```

PHP

Le `break` avec un nombre permet de sortir de plusieurs niveaux de structure de contrôle imbriqués, pas seulement du `switch`

```
$j = 1;
switch ($j) {
    case 1:
        while (true) {
            echo "Dans la boucle et le switch\n";
            break 2; // quitte le while ET le switch
        }
        echo "Jamais exécuté\n";
        break;
    default:
        echo "Autre cas\n";
}
```

Résultat

```
Dans la boucle et le switch
```

PHP

Exemple avec `continue`

```php
$j = 5;
while ($j > 0) {
    if ($j == 3) {
        $j--;
        continue;
    }
    $j--;
    echo $j . "<br>";
}
```

PHP

Exemple avec `continue`

```php
$j = 5;
while ($j > 0) {
    if ($j == 3) {
        $j--;
        continue;
    }
    $j--;
    echo $j . "<br>";
}
```


Résultat

```
4
3
1
0
```

PHP

Considérons le code suivant

```
for ($i = 0; $i < 5; $i++) {
    for ($j = 0; $j < $i + 1; $j++) {
        echo "*";
    }
    echo "<br>";
}
```

PHP

Considérons le code suivant

```
for ($i = 0; $i < 5; $i++) {
    for ($j = 0; $j < $i + 1; $j++) {
        echo "*";
    }
    echo "<br>";
}
```


Résultat

```
*
**
***
****
*****
```

PHP

Si nous voulions ajouter une condition avec un `break`, dans le deuxième `for`, qui permet de quitter les deux boucles

```
$limit = 6;
$compteur = 0;
for ($i = 0; $i < 5; $i++) {
    for ($j = 0; $j < $i + 1; $j++) {
        echo "*";
        $compteur++;
        if ($compteur == $limit) {
            break;
        }
    }
    echo "<br>";
}
```

PHP

Problème

Malheureusement, le résultat reste inchangé car le `break` permet de quitter seulement la deuxième boucle.

PHP

Problème

Malheureusement, le résultat reste inchangé car le `break` permet de quitter seulement la deuxième boucle.

Solution

Définir un label et utiliser `goto`.

PHP

Solution avec `goto`

```
$limit = 6;
$compteur = 0;
for ($i = 0; $i < 5; $i++) {
    for ($j = 0; $j < $i + 1; $j++) {
        echo "*";
        $compteur++;
        if ($compteur == $limit) {
            goto fin;
        }
    }
    echo "<br>";
}
fin:
echo "<br>fin";
```

PHP

Remarque

`break`, `goto` et `continue` rendent le code difficile à lire, à l'exception de l'utilisation de `break` dans `switch`, ils ne doivent pas être utilisés.