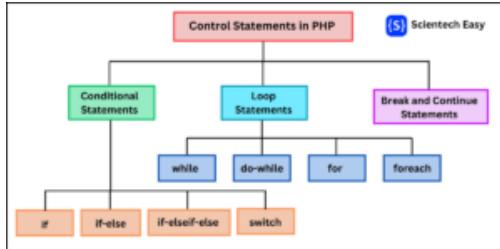


Les boucles et l'instruction switch en PHP



En PHP, les boucles et l'instruction `switch` sont des structures de contrôle essentielles pour gérer le flux d'exécution des programmes. Voici une explication détaillée de ces concepts accompagnée d'exemples pratiques.

1. Les Boucles en PHP

PHP While Loop

```
<html>
<body>
<?php
$i = 1;
while ( $i<=5)
{
echo($i. "<br>");
$i++;
}
?>
</body>
</html>
```

www.educba.com

This block shows a PHP script example for a while loop. It prints numbers from 1 to 5, each on a new line, using the `echo` statement and a counter variable `$i`.

Les boucles permettent d'exécuter un bloc de code plusieurs fois, en fonction d'une condition donnée. PHP propose plusieurs types de boucles :

a. La boucle `for`

Utilisée lorsque le nombre d'itérations est connu à l'avance.

Syntaxe :

```
for (initialisation; condition; incrémentation) {
    // Code à exécuter
}
```

Exemple :

```
<?php
for ($i = 0; $i < 5; $i++) {
```

```
    echo "La valeur de i est : $i\n";
}
?>
```

Sortie :

```
La valeur de i est : 0
La valeur de i est : 1
La valeur de i est : 2
La valeur de i est : 3
La valeur de i est : 4
```

b. La boucle while

Utilisée lorsque le nombre d'itérations n'est pas connu à l'avance et dépend d'une condition.

Syntaxe :

```
while (condition) {
    // Code à exécuter
}
```

Exemple :

```
<?php
$i = 0;
while ($i < 5) {
    echo "La valeur de i est : $i\n";
    $i++;
}
?>
```

Sortie :

```
La valeur de i est : 0
La valeur de i est : 1
La valeur de i est : 2
La valeur de i est : 3
La valeur de i est : 4
```

c. La boucle do...while

Semblable à `while`, mais le bloc de code est exécuté au moins une fois avant que la condition ne soit évaluée.

Syntaxe :

```
do {
    // Code à exécuter
} while (condition);
```

Exemple :

```
<?php
$i = 0;
do {
    echo "La valeur de i est : $i\n";
    $i++;
} while ($i < 5);
?>
```

Sortie :

```
La valeur de i est : 0
La valeur de i est : 1
La valeur de i est : 2
La valeur de i est : 3
La valeur de i est : 4
```

d. La boucle foreach

Spécialement conçue pour parcourir les tableaux et les objets.

Syntaxe :

```
foreach ($tableau as $valeur) {
    // Code à exécuter
}

foreach ($tableau as $clé => $valeur) {
    // Code à exécuter
}
```

Exemple :

```
<?php
$fruits = array("Pomme", "Banane", "Orange");

foreach ($fruits as $fruit) {
    echo "Fruit : $fruit\n";
}
?>
```

Sortie :

```
Fruit : Pomme
Fruit : Banane
Fruit : Orange
```

2. L'instruction switch en PHP

The diagram illustrates the conversion of a nested if-else block into a switch statement. On the left, a block of code uses a series of nested if statements to determine the month name based on a month number. An arrow points from this code to the right, where the same logic is implemented using a switch statement. The switch statement uses the month number as the expression and falls through cases for months 1 through 11, each mapping to a specific month name. A default case handles the case where the month number does not correspond to any known month.

```
var mois = 12;
var moisTexte = "";
if(mois==1){
    moisTexte="Janvier";
} else if(mois==2){
    moisTexte="Février";
} else if(mois==3){
    moisTexte="Mars";
} else if(mois==4){
    moisTexte="Avril";
}
...
else if(mois==12){
    moisTexte="Décembre";
}

var mois = 12;
var moisTexte = "";
switch(mois){
    case 1 : moisTexte="Janvier";
    break;
    case 2 : moisTexte="Février";
    break;
    case 3 : moisTexte="Mars";
    break;
    case 4 : moisTexte="Avril";
    break;
    ...
    case 12 : moisTexte="Décembre";
    break;
    default : console.log("Le mois n'existe pas");
}
```

L'instruction **switch** permet de sélectionner l'exécution d'un bloc de code parmi plusieurs options basées sur la valeur d'une expression.

Syntaxe :

```
switch (expression) {
    case valeur1:
        // Code à exécuter si expression == valeur1
        break;
    case valeur2:
        // Code à exécuter si expression == valeur2
        break;
    ...
    default:
        // Code à exécuter si aucune des valeurs ne correspond
```

}

Important : L'utilisation de `break` est cruciale pour éviter l'**effet de fall-through**, où plusieurs cases sont exécutées successivement.

Exemple :

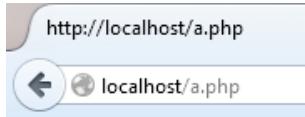
```
<?php
$jour = "Mercredi";

switch ($jour) {
    case "Lundi":
        echo "Aujourd'hui, c'est Lundi.\n";
        break;
    case "Mardi":
        echo "Aujourd'hui, c'est Mardi.\n";
        break;
    case "Mercredi":
        echo "Aujourd'hui, c'est Mercredi.\n";
        break;
    case "Jeudi":
        echo "Aujourd'hui, c'est Jeudi.\n";
        break;
    case "Vendredi":
        echo "Aujourd'hui, c'est Vendredi.\n";
        break;
    default:
        echo "C'est le week-end !\n";
}
?>
```

Sortie :

Aujourd'hui, c'est Mercredi.

3. Combinaison de Boucles et switch



Il est courant de combiner des boucles et des instructions **switch** pour créer des programmes plus complexes. Voici un exemple où nous parcourons un tableau de notes et attribuons une mention en fonction de la note.

Exemple :

```
<?php
$notes = array(85, 62, 74, 90, 58);

foreach ($notes as $note) {
    echo "Note : $note - Mention : ";
    switch (true) {
        case ($note >= 85):
            echo "Excellent\n";
            break;
        case ($note >= 70):
            echo "Bien\n";
            break;
        case ($note >= 60):
            echo "Assez Bien\n";
            break;
        default:
            echo "Insuffisant\n";
    }
}
```

```
}
```

```
?>
```

Sortie :

```
Note : 85 - Mention : Excellent
Note : 62 - Mention : Assez Bien
Note : 74 - Mention : Bien
Note : 90 - Mention : Excellent
Note : 58 - Mention : Insuffisant
```

Explication :

- Nous parcourons le tableau `$notes` avec une boucle `foreach`.
 - Pour chaque note, nous utilisons un `switch` basé sur des conditions (`true` est utilisé pour évaluer des conditions booléennes).
 - En fonction de la valeur de la note, une mention est attribuée.
-

4. Conseils et Bonnes Pratiques

```
# Condition is made $true inside parenthesis
# Hence loop will run endlessly
# As it is never going to be $False

$i = 0
do{
    $i
    $i++;
}while($true)
```

- **Utiliser `break` dans `switch`** : N'oubliez pas d'ajouter `break` après chaque `case` pour éviter l'exécution non désirée des blocs suivants.
- **Boucles infinies** : Soyez prudent avec les boucles `while` et `do...while` pour éviter les boucles infinies. Assurez-vous que la condition de terminaison sera atteinte.
- **Lire la documentation** : PHP offre de nombreuses fonctionnalités avancées pour les boucles et les instructions conditionnelles. Consulter la documentation officielle de PHP peut vous aider à approfondir vos connaissances.

Conclusion

En maîtrisant les boucles et l'instruction `switch` en PHP, vous serez en mesure de créer des scripts dynamiques et réactifs adaptés à une multitude

de situations.