



Lycée Technique de Taza



شهادة التقني العالي

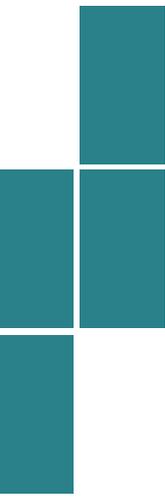
Brevet de Technicien Supérieur

Multimédia et Conception Web

Module : Développement Multimédia

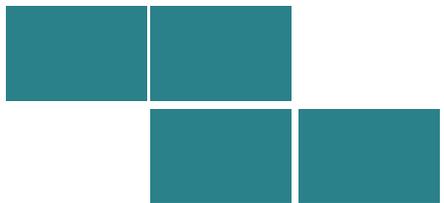
Année professionnelle : 2024 / 2025

Professeur : HICHAM ZAARAOUI



Chapitre 4

Les structures de contrôles



I. Les instructions conditionnelles

○ L'instruction if

L'instruction **if** exécute le code si une **condition est vraie**.

Syntaxe:

```
if (condition) {  
    // code to be executed if condition is true;  
}
```

Exemple :

```
<?php  
$x=18;  
if ($x>3) {  
    echo "La condition est vraie";  
}  
?>
```

I. Les instructions conditionnelles

○ L'instruction if...else

L' instruction **if...else** exécute du code si une condition est **vraie** et un autre code si cette condition est **fausse**.

Syntaxe:

```
if (condition) {  
    // code to be executed if condition is true;  
} else {  
    // code to be executed if condition is false; }
```

Exemple :

```
<?php  
$sexe = "H";  
if ($sexe=="H") {  
    echo "C'est un homme";  
}else{  
    echo "C'est une femme"; }  
?>
```

I. Les instructions conditionnelles

○ L'instruction `if...else`: opérateur ternaire ?

L'opérateur ternaire permet de remplacer l'instruction `if...else`, en attribuant à une variable une 1^{er} valeur si la condition est vraie ou une autre valeur si la condition est fausse.

Syntaxe: `$var= (expression) ?valeur1 :valeur2`

Exemple :

```
<?php
$age=18;
$ch= ($age>=18) ? "Majeure" : "Mineure";
echo $ch;
?>
```

I. Les instructions conditionnelles

○ L'instruction `if...else`: (syntaxe alternative)

La syntaxe alternative permet de clarifier le code, en particulier quand le code PHP est imbriqué avec le code HTML.

Syntaxe :

```
<?php if (condition) : ?>
```

```
<!--code HTML-->
```

```
<?php else: ?>
```

```
<!--code HTML-->
```

```
<?php endif; ?>
```

Exemple :

```
<?php $age=14; if ($age>=18) : ?>
```

```
<p>La personne est majeure</p>
```

```
<?php else: ?>
```

```
<p>La personne est mineure</p>
```

```
<?php endif; ?>
```

I. Les instructions conditionnelles

○ L'instruction `if....elseif....else`:

L'instruction `if....elseif....else` exécute des codes différents pour plus de deux conditions.

Syntaxe:

```
if (condition) {  
  //code to be executed if this  
  condition is true;  
} elseif (condition) {  
  // code to be executed if  
  first condition is false and  
  this condition is true;  
} else {  
  // code to be executed if all  
  conditions are false; }  

```

Exemple:

```
<?php  
$t = date("H");  
if ($t < "10") {  
  echo "Have a good morning!";  
} elseif ($t < "20") {  
  echo "Have a good day!";  
} else {  
  echo "Have a good night!"; }  
?>
```

I. Les instructions conditionnelles

○ L'instruction `switch...case`: (structure de choix)

L'instruction `switch` est utilisée pour éviter la multiplication des instructions `if`.

Syntaxe:

```
switch (expression) {  
    case valeur1: // code à  
exécuter si l'expression correspond  
à valeur1  
    break;  
    case valeur2:// code à  
exécuter si l'expression correspond  
à valeur2  
    break; // ...  
    default: // code à exécuter  
si l'expression ne correspond à  
aucune des valeurs précédentes  
}
```

Exemple:

```
<?php  
$jour = "mercredi";  
switch ($jour) {  
    case "lundi": echo "C'est le  
début de la semaine!"; break;  
    case "mercredi": echo "C'est le  
milieu de la semaine!"; break;  
    case "vendredi": echo "C'est  
presque le week-end!"; break;  
    default: echo "C'est un autre  
jour."; }  
?>
```

I. Les instructions conditionnelles

○ L'instruction **if imbriquée**: (if dans une autre if)

Vous pouvez avoir des instructions **if** à l'intérieur d'autres d'instructions **if**, c'est ce qu'on appelle des *instructions imbriquées if*

Exemple:

```
<?php
$var1=10; $var2=20;
if ($var1>5) {
    if ($var2<30) {
        echo " Les deux conditions sont vraies.";
    }else{
        echo "La 1er condition est vraie,mais la 2eme est fausse.";
    }
} else{
    echo "La première condition est fausse.";
}
?>
```

II. Les structures répétitives en PHP

- La boucle **for** :

La boucle **for** parcourt un bloc de code un nombre de fois **déterminé**.

Syntaxe

```
for (expression1; expression2; expression3) {  
    // code block  
}
```

Exemple

```
<?php  
for ($x=0; $x<=10; $x++) {  
    echo "Le nombre est : $x <br>";  
}  
?>
```

II. Les structures répétitives en PHP

○ La boucle `foreach` :

La boucle `foreach` parcourt un bloc de code pour chaque élément d'un tableau.

Syntaxe:

```
foreach ($tableau as $cle=>$valeur) {  
    // code à executer  
}
```

Exemple 1

```
<?php  
$colors = array("red", "green", "blue", "yellow");  
foreach ($colors as $x) {  
    echo "$x <br>"; }  
?>
```

Exemple 2

```
<?php  
$members = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");  
foreach ($members as $x => $y) {  
    echo "$x : $y <br>"; }  
?>
```

II. Les structures répétitives en PHP

○ La boucle `for` imbriquée :

La boucle `for` imbriquée signifie qu'on utilise une boucle `for` à l'intérieur d'une autre boucle `for` pour parcourir des structures de données complexes comme les tableaux à 2D (matrice).

Syntaxe

```
for (expression1; expression2; expression3) {  
    for (expression4; expression5; expression6) {  
        //code block1  
    }  
    //code block2  
}
```

Exemple

```
<?php  
for ($i=0; $i<3; $i++) { // Boucle externe  
    for ($j=0; $j<2; $j++) { // Boucle interne  
        echo "i = $i, j = $j <br>";  
    }  
}  
?>
```

II. Les structures répétitives en PHP

○ La boucle `while` :

La boucle `while` permet d'exécuter un bloc de code tant qu'une condition donnée est vraie.

Syntaxe:

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

Exemple:

```
<?php  
$compteur=1;  
while ($compteur<=5) {  
    echo "Le compteur est à : $compteur <br>";  
    $compteur++;  
}  
?>
```

Remarque:

Eviter les boucles infinies avec la boucle `while` .

II. Les structures répétitives en PHP

○ La boucle `do....while` :

La boucle `do...while` est similaire à la boucle `while`, à la différence près que la condition est évaluée après l'exécution du bloc de code. Cela signifie que le bloc de code sera exécuté au moins une fois, même si la condition est fausse dès le départ.

Syntaxe:

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

Exemple:

```
<?php  
$compteur = 1;  
do {  
    echo "Le compteur est à : $compteur\n";  
    $compteur++;  
}while ($compteur<= 5);  
?>
```

III. La déclaration de la rupture et de continue

○ break et continue :

□ L'instruction **break** est utilisée pour sortir de la boucle.

Exemple 1:

```
<?php
$i=1;
while ($i<6) {
    if ($i == 3) break;
    echo $i."<br>";
    $i++; }
?>
```

□ L'instruction **continue** est utilisée pour interrompre une iteration dans la boucle.

Exemple 2:

```
<?php
$i=0;
while ($i<6) {
    $i++;
    if ($i == 3) continue; echo $i; }
?>
```

III. La déclaration de la rupture et de continue

Remarques

- ❑ Avec l'instruction **break**, seule la boucle en cours se termine, et pas le script qui se termine par exit.
- ❑ Pour arrêter plusieurs boucles en même temps, on utilise la syntaxe suivante **break n**, où n désigne le nombre de boucles les plus internes.
- ❑ Les mêmes remarques précédentes s'appliquent sur l'instruction **continue**.

VI. Travaux pratiques

Exercice 1

Écrire un script qui vérifie si un nombre donné est pair ou impair.

Exercice 2

Écrire un script qui compare deux nombres et affiche lequel est le plus grand, ou s'ils sont égaux.

Exercice 3

Écrire un script qui prend une note d'un étudiant et affiche le grade correspondant :

- 90-100 : A
- 80-89 : B
- 70-79 : C
- 60-69 : D
- Moins de 60 : F

VI. Travaux pratiques

Exercice 4

Écrire un script qui affiche la table de multiplication de 5 (de 1 à 10) en utilisant une boucle **for**

Exercice 5

Écrire un script qui calcule la somme des nombres de 1 à 100 en utilisant une boucle **while**

Exercice 6

Écrire un script qui demande à l'utilisateur d'entrer un mot de passe jusqu'à ce que le mot de passe correct soit saisi (par exemple : "1234").

VI. Travaux pratiques

Exercice 7

Écrire un script qui affiche tous les nombres pairs entre 1 et 50 en utilisant une boucle **for** et une condition **if**

Exercice 8

Écrire un script qui parcourt un tableau de nombres et qui s'arrête dès qu'il trouve un nombre donné (par exemple : 10).

Exercice 9

Écrire un script qui affiche un triangle d'étoiles en utilisant une boucle imbriquée.