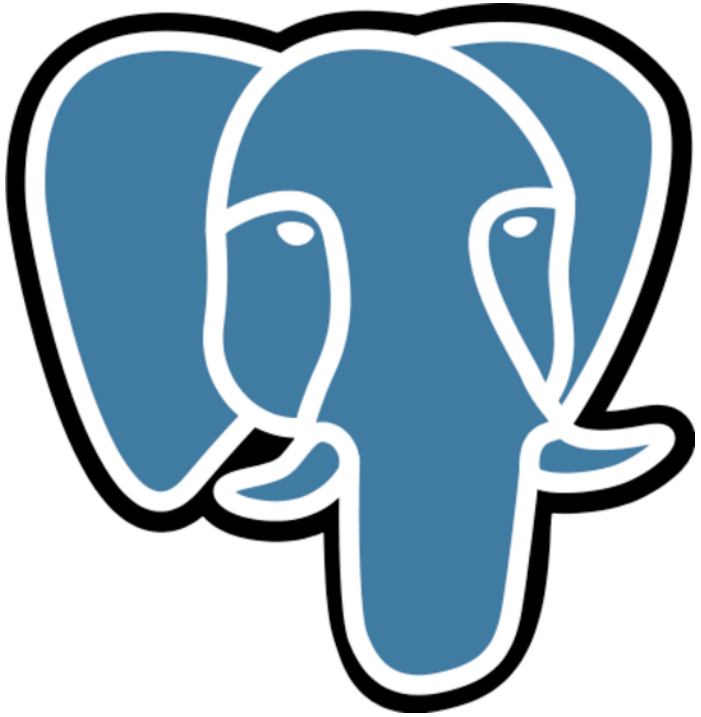




PHP

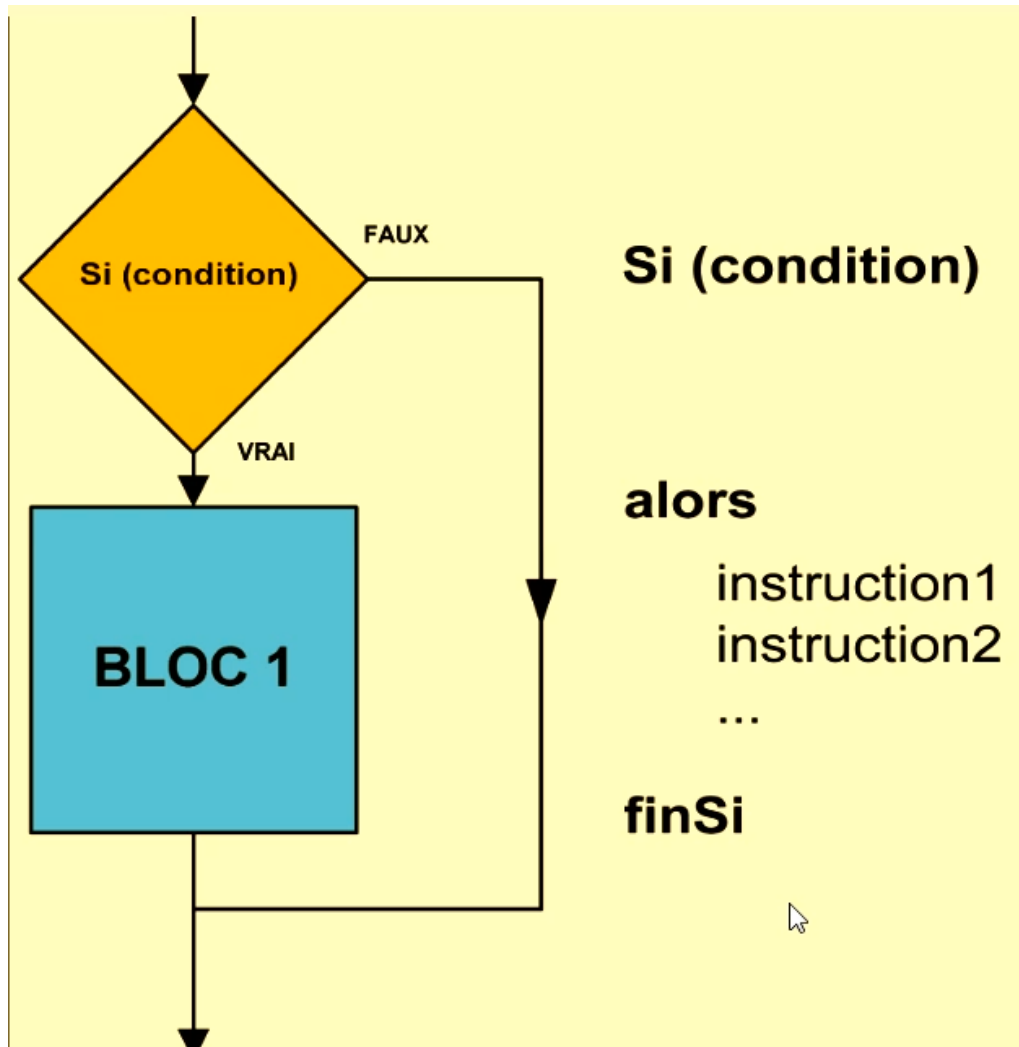
Les structures de programmes



Les tests

Le test conditionnel

Un bloc est un ensemble d'instruction. Il est délimité par des accolades { }



Si (condition)

alors

instruction1
instruction2
...

finSi

En php

```
<?php
$age=23;
if ($age>=18)
{
    echo "Vous êtes majeur";
}
?>
```

Lorsqu'il y a qu'une instruction on peut ne pas mettre les accolades.

Remarque : on peut avoir un **Si** sans avoir forcément un **Sinon** !

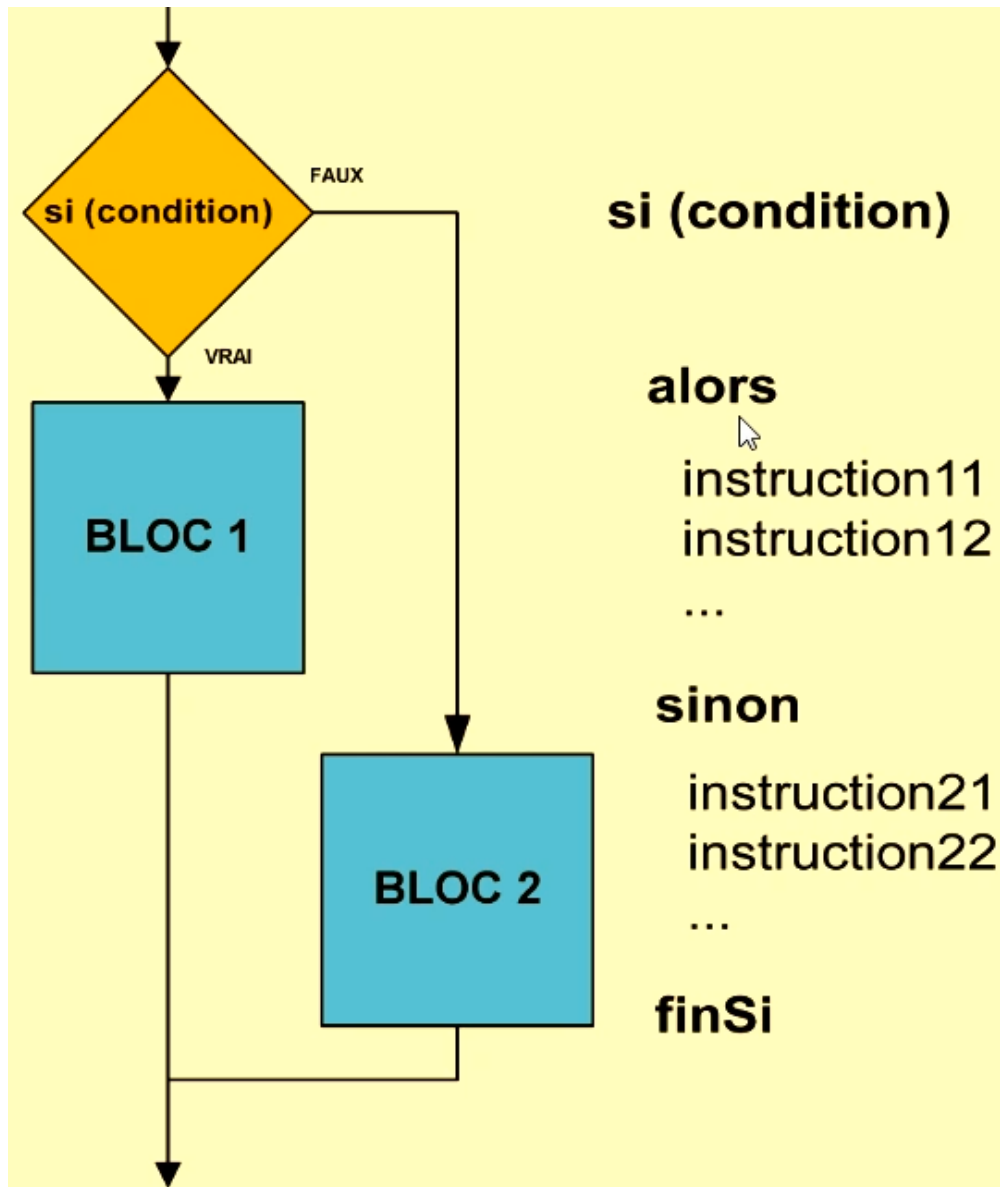
Nature des tests

- On peut utiliser les opérateurs logiques et les combiner

Exemple :

```
if ($age>18 && $age<30) echo «Jeune adulte !»;
```

Test conditionnel avec sinon

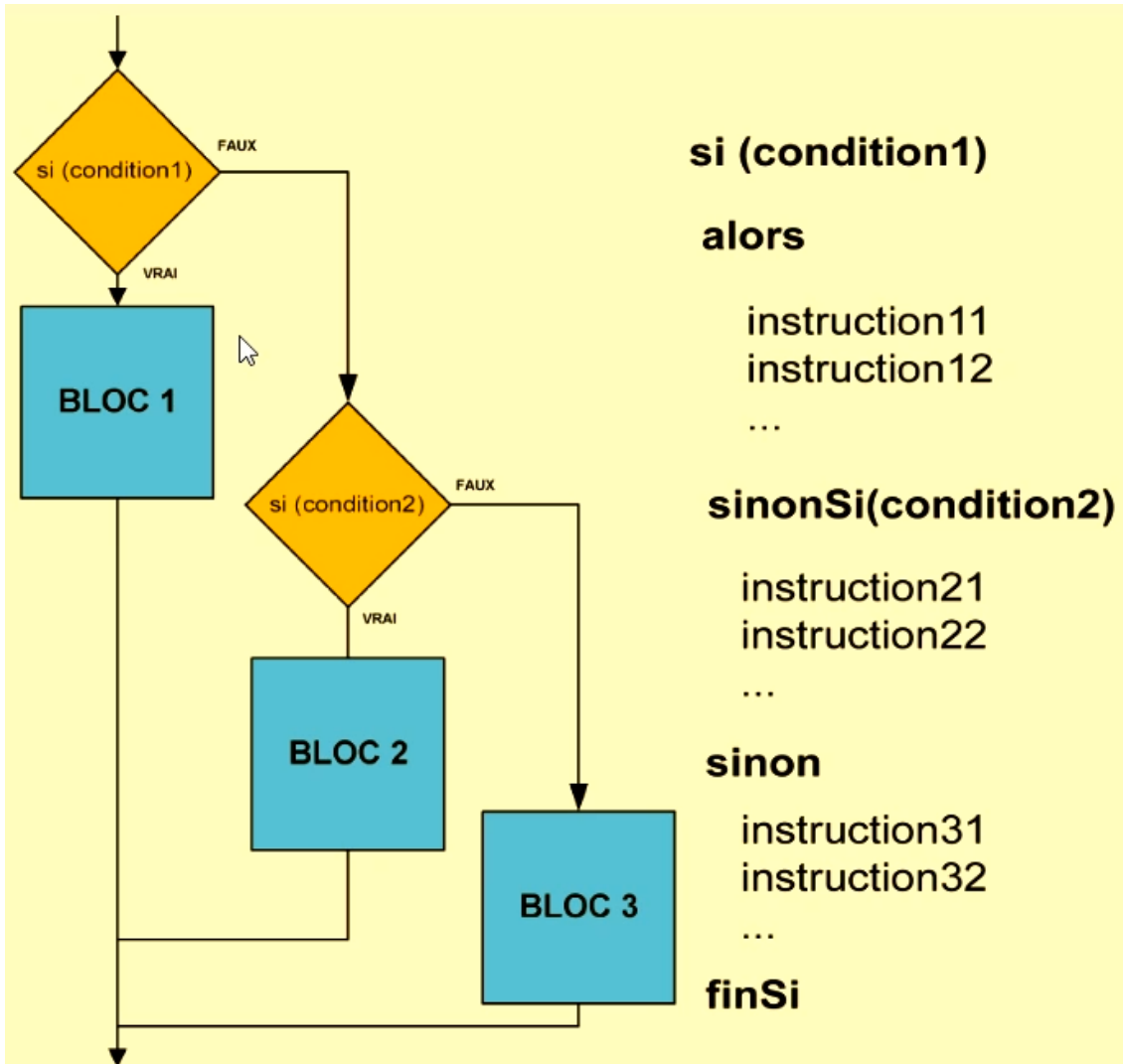


En php

```
<?php
$age=15;
if ($age>=18)
{
    echo "Vous êtes majeur";
}
else
{
    echo "Vous êtes mineur";
}
?>
```

Remarque : L'indentation facilite la lecture ou relecture du code.

Test conditionnel imbriqué

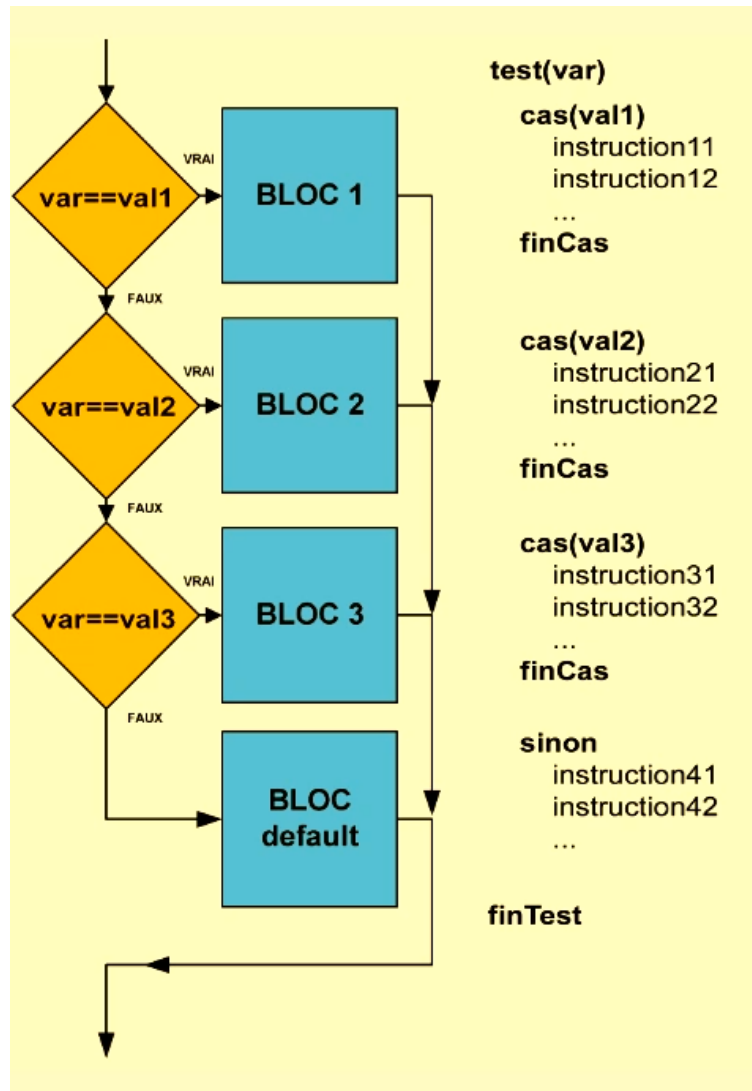


Attention le test d'égalité c'est ==

En php

```
<?php
$age=21;
$sexe="F";
if ($age<18)
{
    echo "Accès interdit !";
}
elseif($sexe=="M")
{
    echo "Soyez le bienvenu";
}
else
{
    echo "Soyez la bienvenue";
}
?>
```

Test avec beaucoup de valeur (switch-case)



En php

```
<?php  
$lg="de";  
switch($lg)  
{  
    case "fr":  
        echo "Bonjour";  
        break;  
    case "en":  
        echo "Hello";  
        break;  
    case "es":  
        echo "Hola";  
        break;  
    case "de":  
        echo "Guten Tag";  
        break;  
    default:  
        echo "Langue inconnue";  
}  
?>
```

Sans le break une fois qu'on rentre dans un bloc : on exécute tous les suivant

Attention : Ne fonctionne qu'avec le signe ==
La dernier bloc (default) est exécuté si aucune condition n'a été vraie avant !
Default est facultatif.



Les boucles

Très important
d'initialiser le
compteur et de
la faire évoluer
pour sortir de la
boucle

Boucle While

En php

initialisation
COMPTEUR

COMPTEUR=nbreTour

boucleSi(condition)

FAUX

VRAI

BLOC 1

évolution
COMPTEUR

boucleSi(condition)

instruction1
instruction2
...

COMPTEUR=COMPTEUR-1

finBoucle

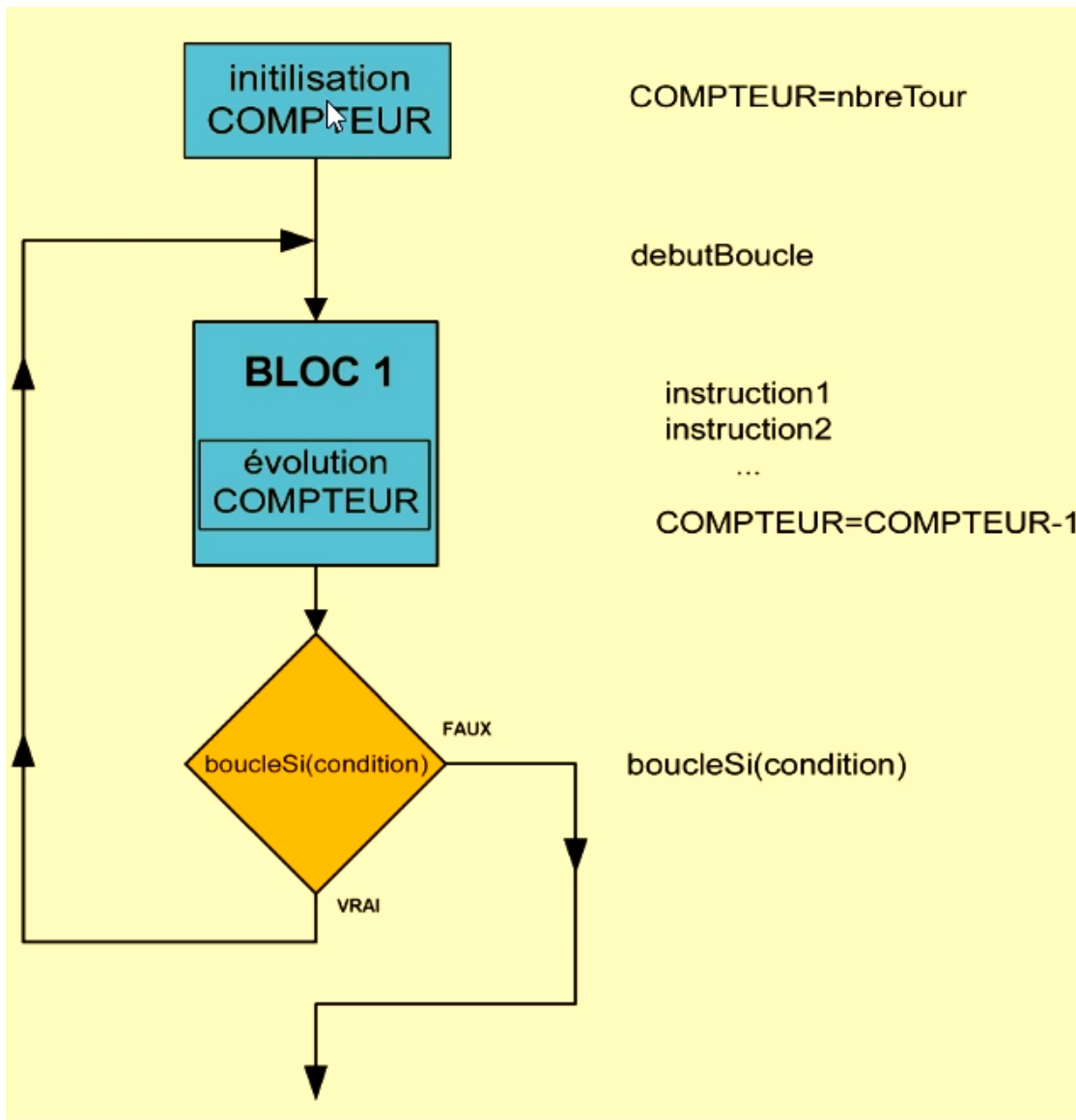
On peut
incrémenter ou
décrémenter le
compteur selon
le test.

```
1 <?php
2 // On initialise le compteur
3 $i=1;
4 // On rentre dans la boucle si la valeur de la variable i
5 // est inférieure à 10
6 // On répète tant que $i <=10
7 while ($i<=10)
8 {
9     echo $i."<br>";
10    // On change la valeur de i
11    // Ici on augmente de 1
12    $i++;
13 }
14 echo "Fin de la boucle";
15 ?>
```

Affichage

1
2
3
4
5
6
7
8
9
10
Fin de la boucle

Boucle do While



En php

```
<html>
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body>
  <?php
    $i = 1;
    do {
      echo $i."<br>";
      $i++;
    }
    while ($i <= 10);
  ?>
</body>
</html>
```

Avec do-while le bloc est exécuté au moins une fois !

Affichage

1
2
3
4
5
6
7
8
9
10
Fin de la boucle

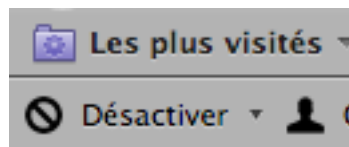
Boucle do While

En php

```
<?php
// On initialise la variable i avec une valeur aléatoire
// Valeur comprise entre 1 et 20
$i=rand(1,20);
// On rentre dans la boucle sans condition
do
{
    echo $i."<br>";
    $i++;
}
while ($i<=10) // Si la condition est vraie on recommence la boucle ({ après le do)
?>
```

Avec do-while le bloc est exécuté
au moins une fois !

Affichage



14

Ici 14 est > à 10 mais comme
le test est effectué après on a
l'affichage !

Boucle For

Même principe que la boucle while mais avec une forme plus compacte.

On utilise la boucle For quand on connaît le nombre d'itération à effectuer !

En php

```
<?php
for ($i=5;$i>0;$i--)
// initialisation ( $i=5 ) et condition $i>0 et décrémentation $i--
{
    echo $i."<br>";
}
?>
```

Boucles imbriquées

On peut imbriquer les boucles : while, do while ou for

Exemple avec For

```
<body>
  <table border="1">
    <?php
      for ($lig=0;$lig<5; $lig++)
      {
        // début de ligne
        echo "<tr>";
        for ($col=0;$col<5;$col++)
        {
          echo "<td>";
          echo $lig."-".$col;
          echo "</td>";
        }
      }
    ?>
  </table>
</body>
```

Résultat



0-0	0-1	0-2	0-3	0-4
1-0	1-1	1-2	1-3	1-4
2-0	2-1	2-2	2-3	2-4
3-0	3-1	3-2	3-3	3-4
4-0	4-1	4-2	4-3	4-4

Boucle foreach

Pour parcourir les tableaux

Php inclut une fonction foreach() qui permet de parcourir un à un les éléments d'un tableau.

Syntaxe :

```
foreach($array as $element) instruction  
foreach($array as $clef=>$element) instruction
```

La deuxième syntaxe permet de récupérer aussi la clef de l'élément d'un tableau associatif.

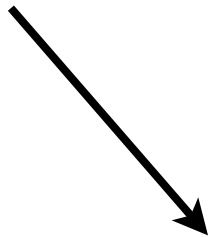
Remarque : foreach() travail sur une copie du tableau et on ne peut donc modifier les valeurs. Si l'on veut modifier les valeurs (passage par référence) il faut faire précéder l'élément de & (depuis PHP 5)

Ex: foreach(\$tab as &\$valeur)

Boucle foreach

Exemples

```
<?php
// Création du tableau et initialisation
$eleve=array('nom'=>'Kan','prénom'=>'Jerry','age'=>16);
// Parcours avec foreach
foreach ($eleve as $clef=>$ele)
{
    echo "La clef ".$clef." le contenu :".$ele."<BR>";
}
?>
```



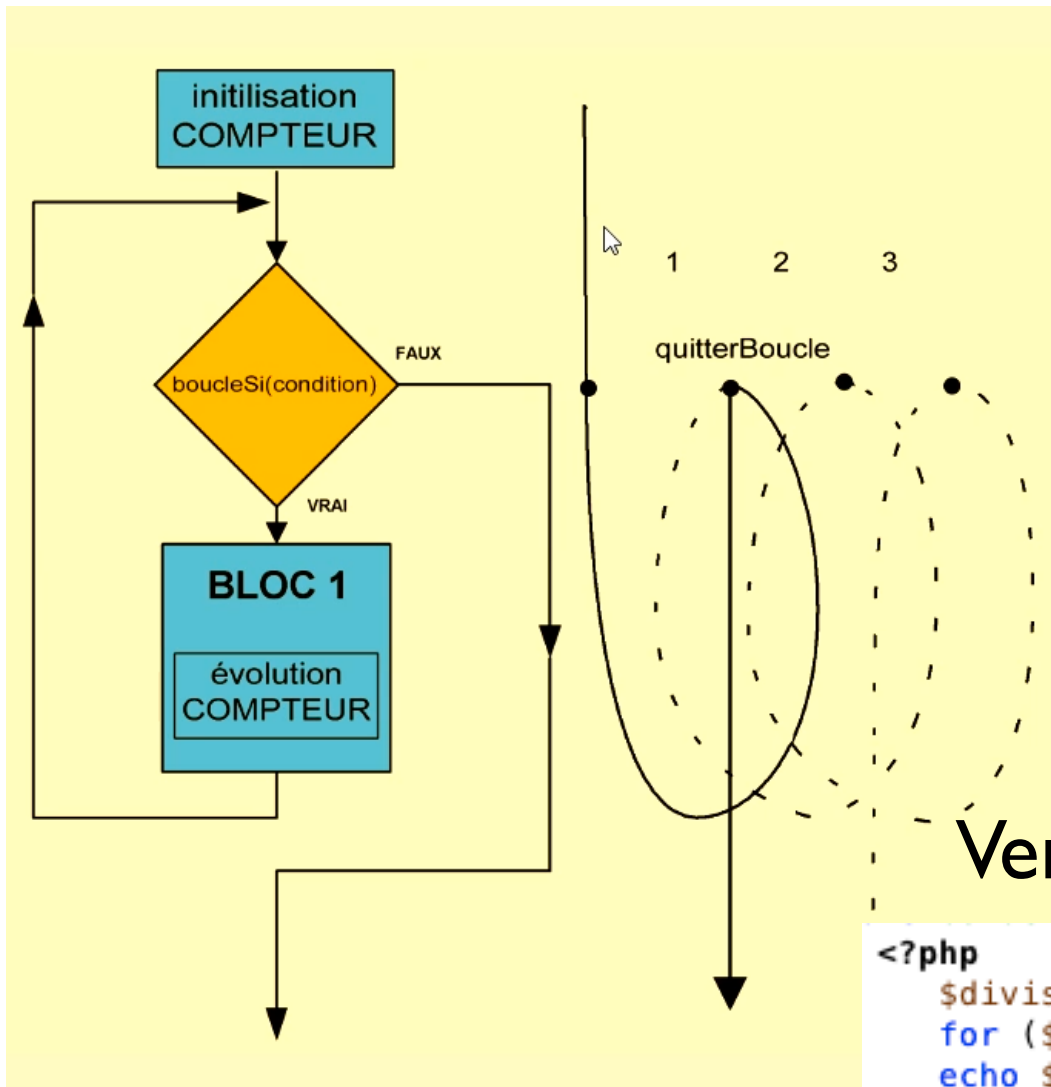
La clef nom le contenu :Kan
La clef prénom le contenu :Jerry
La clef age le contenu :16



Instructions de contrôle

Instruction break

Permet de sortir d'une structure conditionnelle telle que for, while, foreach, switch (déjà vu)



En php

```
<?php

```

Version sans break

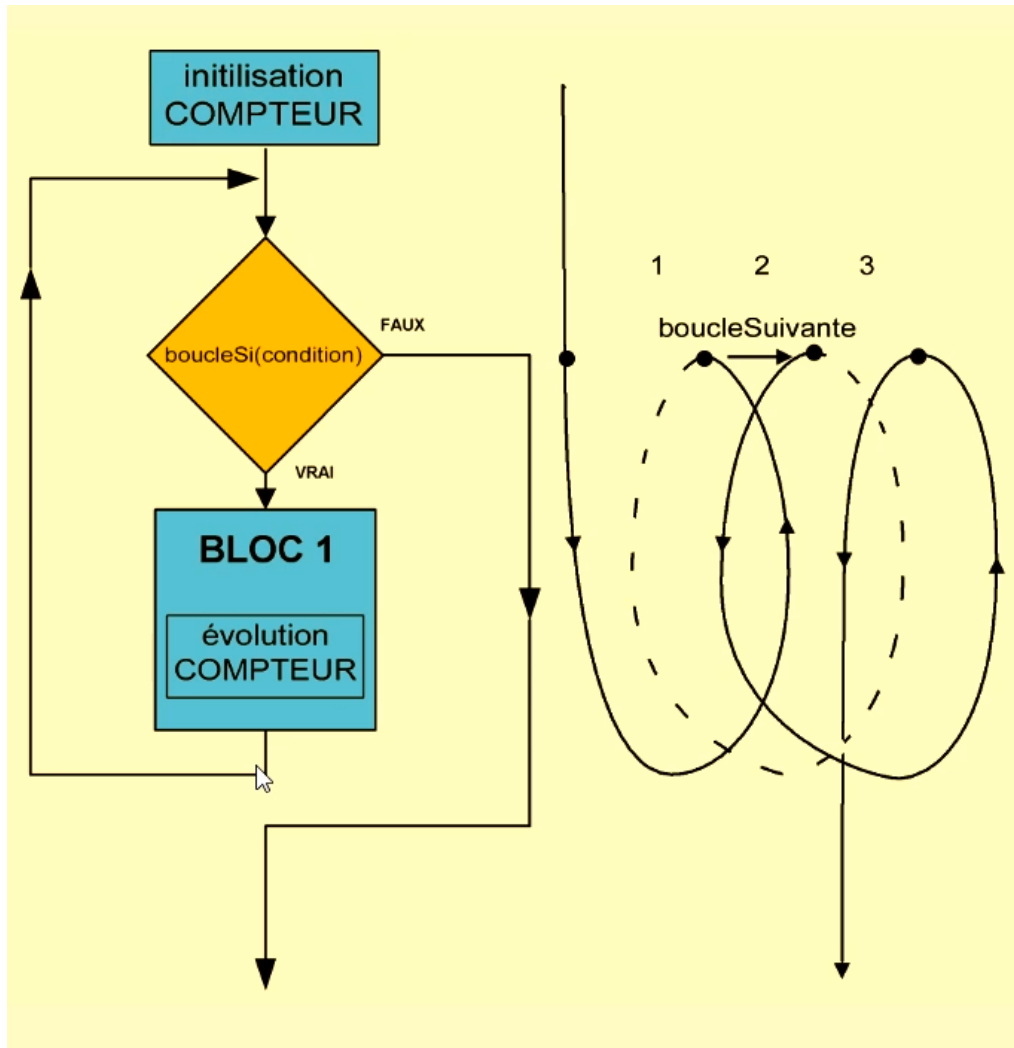
```
<?php

```

1
3
8
8
3
2
fin de boucle

Instruction continue

Permet d'éluder les instructions de l'itération courante et de passer directement à la suivante.



En php

```
<?php

```

1
3
8
8
3
2
2
3
9
12
2
7
fin de boucle