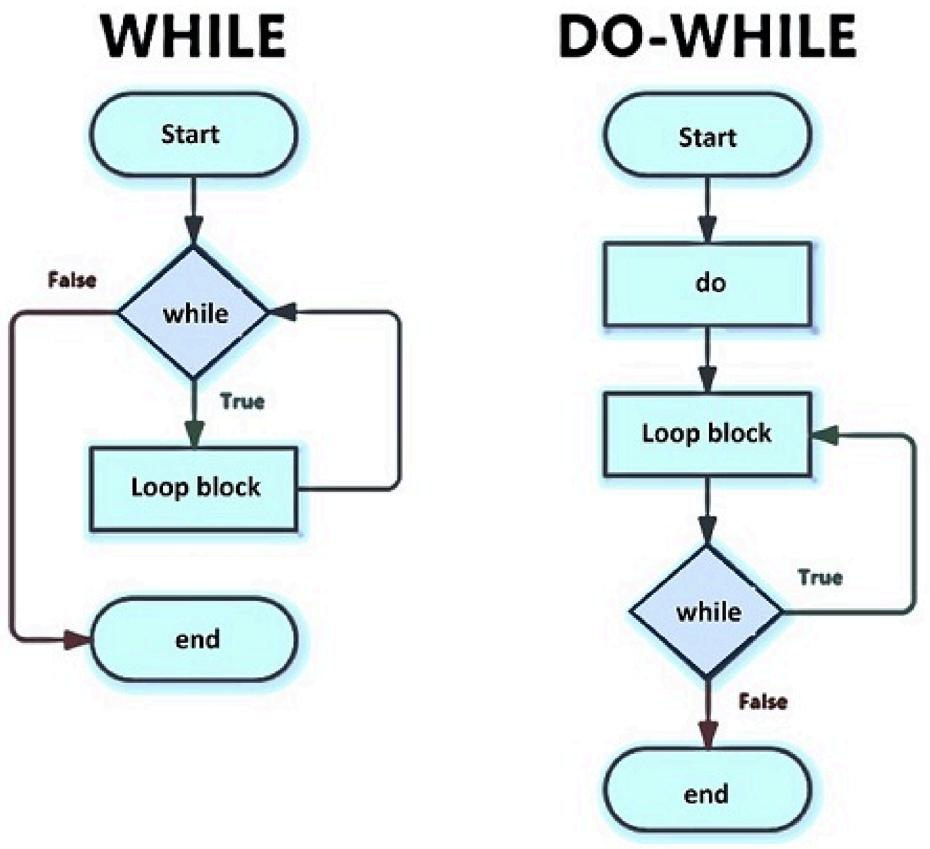
PHP - Do…While Loop

The "do…while" loop is another looping construct available in PHP. This type of loop is similar to the **while** loop, except that the test condition is checked at the end of each iteration rather than at the beginning of a new iteration.

The **while** loop verifies the truth condition before entering the loop, whereas in "do…while" loop, the truth condition is verified before re entering the loop. As a result, the "do…while" loop is guaranteed to have at least one iteration irrespective of the truth condition.

The following figure shows the difference in "while" loop and "do…while" loop by using a comparative flowchart representation of the two.



The **syntax** for constituting a "do…while" loop is similar to its counterpart in C language.

```
do {
   statements;
}
while (expression);
```

Example

Here is a simple example of "do…while" loop that prints iteration numbers 1 to 5.

</> Open Compiler

```
<?php
   $i=1;
   do{
      echo "Iteration No: $i \n";
      $i++;
   }
   while ($i<=5);
?>
```

It will produce the following **output** –

```
Iteration No: 1
Iteration No: 2
Iteration No: 3
Iteration No: 4
Iteration No: 5
```

Example

The following code uses a **while** loop and also generates the same output –

</> Open Compiler

```
<?php
   $i=1;
   while ($i<=5){
      echo "<h3>Iteration No: $i</h3>";
      $i++;
   }
?>
```

Hence, it can be said that the "do…while" and "while" loops behave similarly. However, the difference will be evident when the initial value of the counter variable (in this case **$i**) is set to any value more than the one used in the test expression in the parenthesis in front of the **while** keyword.

Example

In the following code, both the loops – **while** and "**do…while**" – are used. The counter variable for the **while** loop is **$i** and for "do…while" loop, it is **$j**. Both are initialized to 10 (any value greater than 5).

Open Compiler

```php
<?php
   echo "while Loop \n";
   $i=10;
   while ($i<=5){
      echo "Iteration No: $i \n";
      $i++;
   }

   echo "do-while Loop \n";
   $j=10;
   do{
      echo "Iteration No: $j \n";
      $j++;
   }
   while ($j<=5);
?>
```

It will produce the following **output** −

```
while Loop
do - while Loop
Iteration No: 10
```

The result shows that the **while** loop doesn't perform any iterations as the condition is false at the beginning itself (**$i** is initialized to 10, which is greater than the test condition **$i<=5**). On the other hand, the "do…while" loop does undergo the first iteration even though the counter variable **$j** is initialized to a value greater than the test condition.

We can thus infer that the "do…while" loop guarantees at least one iteration because the test condition is verified at the end of looping block. The **while** loop may not do any iteration as the test condition is verified before entering the looping block.

Another syntactical difference is that the **while** statement in "do…while" terminates with semicolon. In case of **while** loop, the parenthesis is followed by a curly bracketed looping block.

Apart from these, there are no differences. One can use both these types of loops interchangeably.

Decrementing a "do…while" Loop

To design a "do…while" with decrementing count, initialize the counter variable to a higher value, use decrement operator (--) inside the loop to reduce the value of the counter on each iteration, and set the test condition in the **while** parenthesis to run the loop till the counter is greater than the desired last value.

Example

In the example below, the counter decrements from 5 to 1.

Open Compiler

```
<?php
   $j=5;
   do{
      echo "Iteration No: $j \n";
      $j--;
   }
   while ($j>=1);
?>
```

It will produce the following **output** –

```
Iteration No: 5
Iteration No: 4
Iteration No: 3
Iteration No: 2
Iteration No: 1
```

Traverse a String in Reverse Order

In PHP, a string can be treated as an indexed array of characters. We can extract and display one character at a time from end to beginning by running a decrementing "do… while" loop as follows −

Open Compiler

```
<?php
   $string = "TutorialsPoint";
   $j = strlen($string);

   do{
      $j--;
      echo "Character at index $j : $string[$j] \n";
   }
   while ($j>=1);
?>
```

It will produce the following **output** −

```
Character at index 13 : t
Character at index 12 : n
Character at index 11 : i
Character at index 10 : o
Character at index 9 : P
Character at index 8 : s
Character at index 7 : l
Character at index 6 : a
Character at index 5 : i
Character at index 4 : r
Character at index 3 : o
Character at index 2 : t
Character at index 1 : u
Character at index 0 : T
```

Nested "do…while" Loops

Like the **for** loop or **while** loop, you can also write nested "do…while" loops. In the following example, the upper "do…while" loop counts the iteration with **$i**, the inner "do… while" loop increments **$j** and each time prints the product of **$i*j**, thereby printing the tables from 1 to 10.

Open Compiler

```php
<?php
   $i=1;
   $j=1;

   do{
      print "\n";
      do{
         $k = sprintf("%4u",$i*$j);
         print "$k";
         $j++;
      }
      while ($j<=10);
      $j=1;
      $i++;
   }
   while ($i<=10);
?>
```

It will produce the following **output** −

```
1   2   3   4   5   6   7   8   9  10
2   4   6   8  10  12  14  16  18  20
3   6   9  12  15  18  21  24  27  30
4   8  12  16  20  24  28  32  36  40
5  10  15  20  25  30  35  40  45  50
6  12  18  24  30  36  42  48  54  60
7  14  21  28  35  42  49  56  63  70
8  16  24  32  40  48  56  64  72  80
9  18  27  36  45  54  63  72  81  90
10  20  30  40  50  60  70  80  90 100
```