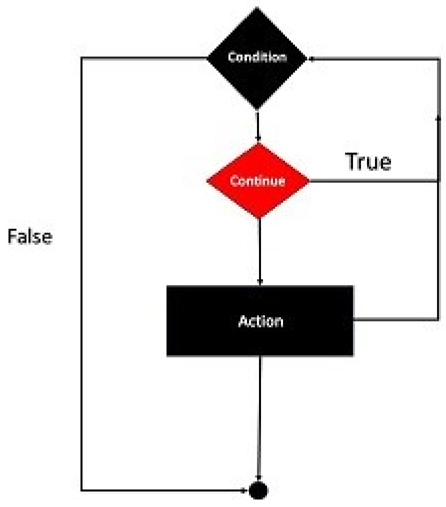
PHP - Continue Statement

Like the **break** statement, **continue** is another "loop control statement" in PHP. Unlike the **break** statement, the **continue** statement skips the current iteration and continues execution at the condition evaluation and then the beginning of the next iteration.

The **continue** statement can be used inside any type of looping constructs, i.e., **for, foreach, while** or **do-while** loops. Like **break**, the **continue** keyword is also normally used conditionally.

```
while(expr){
   if (condition){
      continue;
   }
}
```

The following **flowchart** explains how the **continue** statement works −



Example

Given below is a simple example showing the use of **continue.** The **for** loop is expected to complete ten iterations. However, the **continue** statement skips the iteration whenever the counter id is divisible by 2.

Open Compiler

```
<?php
   for ($x=1; $x<=10; $x++){
      if ($x%2==0){
         continue;
      }
      echo "x = $x \n";
   }
?>
```

It will produce the following **output** –

```
x = 1
x = 3
x = 5
x = 7
x = 9
```

Example

The **continue** statement accepts an optional numeric argument which tells it how many levels of enclosing loops it should skip to the end of. The default is 1.

Open Compiler

```
<?php
   for ($i=1; $i<=3; $i++){
      for ($j=1; $j<=3; $j++){
         for ($k=1; $k<=3; $k++){
            if ($k>1){
               continue 2;
            }
            print "i: $i  j:$j  k: $k\n";
         }
      }
   }
?>
```

It will produce the following **output** –

```
i: 1  j:1  k: 1
i: 1  j:2  k: 1
i: 1  j:3  k: 1
i: 2  j:1  k: 1
```

```
i: 2  j:2  k: 1
i: 2  j:3  k: 1
i: 3  j:1  k: 1
i: 3  j:2  k: 1
i: 3  j:3  k: 1
```

The **continue** statement in the inner **for** loop skips the iterations 2 and 3 and directly jumps to the middle loop. Hence, the output shows "k" as 1 for all the values of "i" and "k" variables.