Equivalence of Looping Structures

This lesson explains how we can covert a for loop into a while loop.

We'll cover the following

- For loop
- Converting For Loop to While Loop
- for loop versus while loop

For loop

An example of `for` loop is given below:

```php
<?php
$i;
for ($i = 0;$i < 10;$i++)
{
    $i = $i * 2;
    echo "Value of i is: $i\n";
}
echo "Final value of i is: $i\n";
?>
```

Converting For Loop to While Loop

The above for loop can easily be rewritten as a while loop. Notice the extra enclosing brackets, and the two extra semicolons after the expressions in order to turn them into statements.

```php
<?php
$i=0 ;
while ($i<10) {
  $i = $i*2;
  echo "Value of i is: $i\n";
  $i++;
}
echo "Final value of i is: $i\n";
?>
```

for loop versus while loop

A `for` loop is more often used by programmers due to its conciseness as well as its separation of the looping logic (often using a loop control variable like **“$i”** or another simple iterator) from the loop’s content.

A `while` loop is often preferred if the *initial* statement or update statement requires more complex code than what would fit neatly into the `for` construct. However, the two are *equivalent*. Therefore, it is ultimately a coding style decision, not a technical decision about whether to use one or the other.

In the next lesson, we’ll discuss infinite loops and how they arise.