Loops

In deze les ga je de for-loop, de while-loop en de do-while loop leren.

Loops komen in alle programmeertalen voor. Zij zorgen ervoor dat je een bepaalde handeling tig keer kunt uitvoeren zonder dat je hele lange en bijna dezelfde code moet intypen.

For-loop

Stel je wil 10x een bepaalde regel printen. Daarvoor kun je de regel in een loop zetten:

```
<?php
for($i=0; $i<10; $i++) {
echo "De waarde van \$i = $i<br>";
}
?>
```

Voor deze code uit en kijk wat er gebeurt.

De for-loop wordt een bepaald aantal keren uitgevoerd. Dit aantal keer stel jij als programmeur vast. De loop kent 3 fases: begin, de uitvoering en het eind.

Begin

De loop begint met $i=0 (dit het eerste argument van de for-loop).
Voordat de loop mag worden uitgevoerd wordt de vergelijking in het tweede argument uitgevoerd; alleen als deze vergelijking true oplevert mag de eerste *iteratie* (ronde) beginnen.

Uitvoering

De loop gaat lopen en bij elke *iteratie* (=volgende ronde) wordt eerst het derde argument uitgevoerd. In dit geval dus $i++. Daarna wordt gekeken of de vergelijking in het tweede argument nog true oplevert. Als dat zo is dan gaat de loop de volgende iteratie in.

Eind

Zodra de vergelijking in het tweede argument false oplevert stopt de loop.

Opdrachten

Schrijf van alle volgende loops op hoe vaak die wordt uitgevoerd, bedenk ook waarom.

1. `for($i=5; $i<=10; $i++){...`
2. `for($i=100; $i<102; $i++){...`
3. `for($i=10; $i>0; $i--){...`
4. `for($i=0; $i<10; $i=$i+2){...`
5. `for($x=0; $i<10; $X--){...`

While-Loop

De while loop is eigenlijk eenvoudiger uit te leggen. Hij kent één vergelijking en zolang die true is blijft de loop doorlopen.

```
$i=0;
while($i<10){
	$i++;
    ...
}
```

Met deze while-loop doe je eigenlijk hetzelfde als in de for loop: `for($i=0; $i<10; $i++)`

Deze loop is iets 'gevaarlijker' omdat de drie fases van de loop niet per sé bij elkaar staan. Dat is ' gevaarlijk' omdat je dan eerder het overzicht verliest en misschien het ophogen (in het voorbeeld regel3) vergeet. Ook kan het minder overzichtelijk zijn.

Bij een while-loop moet je net als bij een for-loop altijd nadenken over de 3 fasen:

1. hoe begint de loop (in het voorbeeld $i=0);
2. wat gebeurt er bij elke iteratie en;
3. hoe stop de loop?

Stel je hebt een functie die een row uit de database haalt, deze functie heet getRow() en deze functie returned de regel. Als er geen regels meer zijn dat wordt er 0 teruggegeven. Je kunt nu een

while-loop gebruiken om alle regels af te drukken:

```
<?php
$dezeRegel = getRow();

while($dezeRegel<>0){
echo $dezeRegel;
    $dezeRegel=getRow();
}
?>
```

Het is gebruikelijk om dit op deze manier uit te voeren, maar het kan ook in een for-loop. Welke methode vind jij beter leesbaar?

```
<?php
for( $dezeRegel = getRow(); $dezeRegel<>0; $dezeRegel=getRow();){
echo $dezeRegel;
}
?>
```

In sommige gevallen wil je eindeloze loop maken. Een while loop kun je eenvoudig eindeloos maken: `while(true){...`

Do-While

De do-while loop is een variant op de while-loop. In deze loop staat de vergelijking aan het eind in plaats van in het begin zoals bij de while-loop en de for-loop.

```
<?php
$i = 0;
do {
    echo $i;
} while ($i > 0);
?>
```

Wat denk je dat deze code doet? Probeer deze code. Als je deze code uitvoert, dan zie je dat de loop 1x uitgevoerd wordt en dat terwijl $i nooit groter dan 0 is (en dus de vergelijking nooit true oplevert). Een Do-While loop wordt dus altijd minimaal 1x uitgevoerd. Dat geldt niet voor de ' gewone' while-loop of voor de for-loop.

Stroomdiagrammen

Hieronder zie je twee stroom-diagrammen. Welke hoort bij de while en welke bij de do-while? Welke zou het beste passen bij een for-loop?

Image result for do while loop php Image not found or type unknown	Image result for while loop in php Image not found or type unknown

Break

Met een break statement spring je uit de huidige loop. Je gaat direct naar het eind en begint met het statement dat vlak na de } staat aan het einde van de loop.

Continue

Met het continue statement stop je met de huidige iteratie en ga je naar het begin van de loop om de volgende iteratie te beginnen.

Revision #8
Created 6 October 2019 13:19:08 by Admin
Updated 4 February 2020 10:06:55 by Max