PHP: programmabesturing

Vertakkingen

Normaal gesproken wordt een programma regel voor regel uitgevoerd door een computer. Dit noemen we **sequentie**. Of wel opdrachten worden sequentieel uitgevoerd door de computer.

Er zijn echter speciale structuren waarmee de programmeur kan bepalen welke instructies de computer moet uitvoeren afhankelijk van vast gestelde voorwaarden. Deze noemen we **selectie** structuren.

Tevens zijn er structuren waar bij de computer een groep opdrachten meerdere keren kan herhalen. Deze worden **iteratie** structuren genoemd.

Ook is het mogelijk om een verzameling instructie herhaaldelijk te kunnen aan roepen met een naam. Dit zijn subroutines of wel **functies**.

Selectie-structuren

De **if-structuur** wordt gebruikt om aan de hand van een expressie te kunnen bepalen welke instructies de computer moet uit voeren. Een expressie geeft een logische waarde terug (TRUE of FALSE). Hieronder zie je de basis syntax voor de if-structuur:

```
if ( expressie/voorwaarde )
        { Instructies die uit gevoerd worden als de expressie waar is (TRUE) }
elseif (expressie)
        { Instructies die uit gevoerd worden als de expressie waar is (TRUE) }
else
        { Instructies die uit gevoerd worden als de expressie niet waar is (FALSE) }
```

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
if ( 4 > 2 ) {echo '<p>Ja, 4 is groter dan 2 <br>';}

if ( ( 4 > 2 ) && ( 8 > 4 ) )
        {echo '4 is groter dan 2 en 8 is groter dan 4 <br>';}

if ( 4 > 8)
        {echo '4 is groter dan 8 <br>';}
else
        {echo '4 is kleiner dan 8 <br>';}

if ( 4 > 8 )
        {echo 'Deze test is Waar </p>';}
elseif ( 8 > 4 )
        {echo 'De alternatieve test is Waar </p>';}
```

```
else
        {echo 'Beide testen zijn Onwaar </p>';}
```

2 Sla het document op in **/htdocs** als **ifelse.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

```
Ja, 4 is groter dan 2
4 is groter dan 2 en 8 is groter dan 4
4 is kleiner dan 8
De alternatieve test is Waar
```

Een andere selectie structuur is de **switch-structuur**. Deze wordt gebruikt ter vervanging van meerdere **if-elsif-else** structuren achter elkaar als de test expressie één conditie moet evalueren.
De **switch** instructie heeft één waarde als parameter en kijkt of deze gelijk is aan één van de waarden in de **case** instructie. De code behorende bij die **case** instructie wordt dan uitgevoerd.
Elke case instructie moet eindigen met een **break** instructie.
Optioneel kan de lijst met case instructies gevolgd worden door een enkele **default** instructie aan het einde. De code die bij de **default** instructie staat wordt dan uitgevoerd als er geen gelijke waarde wordt gevonden in de lijst van **case** instructies.
In de afbeelding hieronder staat de syntax (schrijfwijze) voor de switch instructie:

```
switch ( test-waarde )
{
        case match-waarde:
                instructies die uitgevoerd worden indien waarden gelijk zijn;
                break;
        case match-waarde:
                instructies die uitgevoerd worden indien waarden gelijk zijn;
                break;
        case match-waarde:
                instructies die uitgevoerd worden indien waarden gelijk zijn;
                break;
        default:
        i       nstructies die uitgevoerd worden indien geen waarden gelijk zijn;
}
```

Een switch instructie kan vele case instructies hebben maar er kunnen geen twee case instructies zijn die een zelfde waarde proberen te vergelijken.

PHP: programmabesturing

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
$nummer = 2; $letter = 'B';

switch ( $nummer )
{
	case 1: echo 'Nummer is Eén<br>'; break;
	case 2: echo 'Nummer is twee<br>'; break;
	case 3: echo 'Nummer is Drie<br>'; break;
	default: echo 'Nummer is niet herkend<br>'; break;
}

switch ( $letter )
{
	case 'A' : echo 'Letter is A<br>'; break;
	case 'B' : echo 'Letter is B<br>'; break;
	case 'C' : echo 'Letter is C<br>'; break;
	default : echo 'Letter is niet herkend<br>';
}

switch ( $nummer )
{
	case 0 : case 1 : case 2 : echo 'Minder dan 3<br>'; break;
	default : echo '3 of meer, of minder dan nul';
}
```

2 Sla het document op in **/htdocs** als **switch.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

Nummer is twee
Letter is B
Minder dan 3

Iteratie structuren

Een iteratie, vaak een lus of loop genoemd, is code in een script dat automatisch herhaalt wordt. Eén volledige uitvoering van alle instructies binnen de lus wordt een iteratie genoemd.
Het aantal keren dat de instructies in een lus herhaalt wordt bepaald een expressie die een voorwaarde (**criteria**) bevat. Zolang de voorwaarde blijft gelden, waar is (**TRUE**), zal de lus blijven herhalen. Zodra de voorwaarde van de expressie niet meer geldig is, niet waar (**FALSE**), stopt de lus.

PHP: programmabesturing

In PHP zijn er vier verschillende lus-structuren:

- **for** lus
  - Deze voert een specifiek aantal iteraties uit.
- **while** lus
  - Deze voert iteraties uit zolang een test van de criteria aan het begin van iedere iteratie de waarde TRUE geeft.
- **do - while** lus
  - Deze voert iteraties uit zolang een test van de criteria aan het einde van iedere iteratie de waarde TRUE geeft.
- **foreach** lus
  - Deze voert een iteratie uit door elk element van een array.

```
for (initialisatie; test-expressie; verhogen/verminderen)
{
        Uit te voeren instructies
}
```

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
for ( $i = 1 ; $i < 4 ; $i++)
{
        echo "<dt> Buitenste lus iteratie $i";

        # Geneste lus
        for ( $j = 1 ; $j < 4 ; $j++)
        {
                echo "<dd> binneste lus iteratie $j";
        }
}
```

2 Sla het document op in **/htdocs** als **forlus.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

Buitenste lus iteratie 1
        binneste lus iteratie 1
        binneste lus iteratie 2
        binneste lus iteratie 3
Buitenste lus iteratie 2
        binneste lus iteratie 1
        binneste lus iteratie 2
        binneste lus iteratie 3
Buitenste lus iteratie 3
        binneste lus iteratie 1
        binneste lus iteratie 2
        binneste lus iteratie 3

PHP: programmabesturing

```
initialistie
while ( test-expressie )
{
        Uit te voeren instructies;
        verhoog/verminder instructie;
}
```

```
initialistie
do
{
        Uit te voeren instructies;
        verhoog/verminder instructie;
} while ( test-expressie );
```

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```php
$nummers = array( 10 , 20 , 30);

echo '<dt> While Lus : ';
$i = 0 ;
while ( $i < 3 )
{
        echo "<dd>Element $i = $nummers[$i]";
        $i++;
}

echo '<dt>Do-While Lus : ';
$i = 0;
do
{
        echo "<dd>Element $i = $nummers[$i]";
        $i++;
} while ( $i < 3 );
```

2 Sla het document op in **/htdocs** als **whilelus.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

```
While Lus :
        Element 0 = 10
        Element 1 = 20
        Element 2 = 30
Do-While Lus :
        Element 0 = 10
        Element 1 = 20
        Element 2 = 30
```

Lussen beëindigen

De **break** instructie kan gebruikt worden om een lus vroegtijdig te beëindigen.

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
for ( $i = 1 ; $i < 4 ; $i++)
{
        for ( $j = 1 ; $j < 4 ; $j++)
        {
                echo "Iteraties i = $i  en j = $j <br>";
        }
}
```

2 Sla het document op in **/htdocs** als **breakcontinue-1.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

```
Iteraties i = 1 en j = 1
Iteraties i = 1 en j = 2
Iteraties i = 1 en j = 3
Iteraties i = 2 en j = 1
Iteraties i = 2 en j = 2
Iteraties i = 2 en j = 3
Iteraties i = 3 en j = 1
Iteraties i = 3 en j = 2
Iteraties i = 3 en j = 3
```

PHP: programmabesturing

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
for ( $i = 1 ; $i < 4 ; $i++)
{
      for ( $j = 1 ; $j < 4 ; $j++)
      {
            if ( $i == 2 && $j == 1)
            {
                echo "Stopt binnenste lus wanneer i = $i  en j = $j <br>";
                break;
            }
            echo "Iteraties i = $i  en j = $j <br>";
      }
}
```

2 Sla het document op in **/htdocs** als **breakcontinue-2.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

```
Iteraties i = 1 en j = 1
Iteraties i = 1 en j = 2
Iteraties i = 1 en j = 3
Stopt binnenste lus wanneer i = 2 en j = 1
Iteraties i = 3 en j = 1
Iteraties i = 3 en j = 2
Iteraties i = 3 en j = 3
```

De instructie continue kan gebruikt worden om een enkele iteratie over te slaan wanneer aan een specifieke voorwaarde wordt voldaan.

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
for ( $i = 1 ; $i < 4 ; $i++)
{
      for ( $j = 1 ; $j < 4 ; $j++)
      {
            if ( $i == 1 && $j == 1)
            {
               echo "Ga door met binnenste lus bij i = $i  en j = $j <br>";
               continue;
            }
            if ( $i == 2 && $j == 1)
            {
                echo "Stopt binnenste lus wanneer i = $i  en j = $j <br>";
                break;
```

```
            }
            echo "Iteraties i = $i  en j = $j <br>";
        }
}
```

2 Sla het document op in **/htdocs** als **breakcontinue-3.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

```
Ga door met binnenste lus bij i = 1 en j = 1
Iteraties i = 1 en j = 2
Iteraties i = 1 en j = 3
Stopt binnenste lus wanneer i = 2 en j = 1
Iteraties i = 3 en j = 1
Iteraties i = 3 en j = 2
Iteraties i = 3 en j = 3
```

Functies maken

PHP heeft vele ingebouwde functies die bij naam kunnen worden aangeroepen, zoals de verschillende array sorteer functies die eerder besproken zijn.
Er kunnen ook eigengemaakte functie gebruikt worden in PHP. Dit zorgt ervoor dat scripts in blokken codes kunnen worden gestructureerd die beter te begrijpen en onderhouden zijn.
Een eigen gemaakte functienaam mag een combinatie van letter, nummers en het underscore '_' teken bevatten, maar moet beginnen met een letter of het underschore teken.
De naam mag niet gelijk zijn aan die van een ingebouwde functie.
Net als met variabelen, zijn functienamen hoofdletter gevoelig.
In de afbeelding hieronder zie je de basis syntax voor het maken van een functie:

```
function functienaam ()
{
        Instructies die uitgevoerd dienen te worden
}
```

Bij het maken van een functie is het belangrijk om te begrijpen hoe de toegang tot een variabele gelimiteerd is, "scope":

- Variabelen die buiten een functie zijn gemaakt, hebben een "globale" scope maar zijn niet toegankelijk vanuit elk functieblok.
- Variabelen gemaakt binnen een functie hebben een "lokaal" bereik en zijn dus alleen toegankelijk vanuit dat specifieke functieblok.

PHP: programmabesturing

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
$resultaat = 5 + 5 ;

function square()
{
	$resultaat = 5 * 5 ;
	echo "Lokaal Vierkant Resultaat = $resultaat<br>";
}

function cube()
{
	$resultaat = 5 * 5 * 5 ;
	echo "Lokaal kubus Resultaat = $resultaat<br>";
}

echo "Globale Som Resultaat = $resultaat<br>";
square();
cube();
```

2 Sla het document op in **/htdocs** als **functie.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

Globale Som Resultaat = 10
Lokaal Vierkant Resultaat = 25
Lokaal kubus Resultaat = 125

Parameters doorgeven

Data kan als "argumenten" of wel parameters worden doorgegeven aan een functie.
Er kunnen verschillende parameters gebruikt worden bij de aanroep van een functie. Echter moet er wel de juiste data mee worden gegeven voor elke parameter. De afbeelding hieronder geeft de syntax weer voor parameter aanroep van een functie:

```
function functienaam ( $arg_1, $arg_2, $arg_3)
{
	Instructies die uitgevoerd dienen te worden
}
```

PHP: programmabesturing

1. Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
$nummer = 5;

function square($nummer)
{
        $resultaat = $nummer * $nummer ;
        echo "$nummer Vierkant = $resultaat<br>";
}

function cube($nummer)
{
        $resultaat = $nummer * $nummer * $nummer ;
        echo "$nummer kubus = $resultaat<br>";
}

square( 3 );
cube( 4 );
square($nummer);
cube($nummer);
```

2. Sla het document op in **/htdocs** als **argument.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

3 Vierkant = 9
4 kubus = 64
5 Vierkant = 25
5 kubus = 125

PHP: programmabesturing

Waarden teruggeven

Sommige PHP functie geven een waarde terug nadat ze aangeroepen zijn. De terug te geven waarden kan bepaald worden door de mee gegeven parameters. De tabel hieronder geeft voorbeelden weer voor de **date()** functie:

Argument:	Omschrijving:	Voorbeeld:
Y	Year as four digits	2018
y	Year as two digits	18
n	Month as one or two digits	11
m	Month as two digits	07
F	Month full name	November
M	Month name as three letters	Nov
j	Day as one or two digits	4
d	Day as two digits	04
l	Weekday full name	Monday
D	Weekday as three letters	Mon
S	Ordinal suffix as two letters	th
H	Hour in 24-hour format	14
i	Minutes	45
s	Seconds	30

1 Maak een correct HTML5 document en tik het volgende script in tussen de PHP tags.

```
$gebruiker = ' Buchi ';
function display_date()
{
	return date( 'l, jS F' );
}
function welcome( $gebruiker )
{
	$hour = date ( 'H');
	$greeting = ( $hour < 12 ) ? 'br>Bon Dia' : '<br>Bon Tardi';
	$greeting .= $gebruiker;
	return $greeting;
}
echo display_date();
echo welcome( $gebruiker );
```

2 Sla het document op in **/htdocs** als **return.php** en open het daarna in je browser. Je ziet dan het onderstaande resultaat.

Friday, 14th September
Bon Tardi Buchi