

## Hoofdstuk 6 : Besturing

Tot nu toe zijn we vooral bezig geweest met het aanleren van begrippen, definities, ... Het element 'actie' ontbreekt. Met 'actie' bedoelen we hier, dat we een stukje programma schrijven die 'iets' uitvoert.

De gebruiker (=website-bezoeker) maakt op het scherm dmv een 'klik' een bepaalde keuze. Op basis daarvan voert een PHP een 'actie' uit. (Voorbeelden: een product-beschrijving weergeven, een foto weergeven, een berekening maken, ....).  
Maakt de gebruiker een 'andere' keuze...dan moet een 'andere' actie volgen.

We beschrijven hier een aantal besturings-mechanismes.

### 6.1 If...else

Waarschijnlijk de meest belangrijke.

('If' kun je vertalen door 'als'....'else' door 'anders')

'Als' (if) iets is vervuld doe je 'a'...'anders' (else) doe je 'b'.

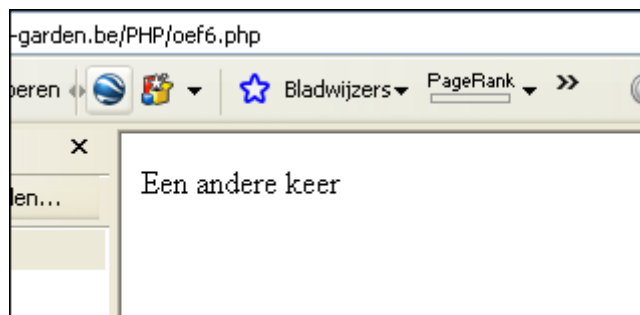
'Als het minder kost dan 10 euro koop ik het, anders niet'

Open start.php, en voeg onderstaande code toe in de <body>:

```
<?php
$prijs=12;
if ($prijs<=10){
echo ("Verkocht");}
else
{echo ("Een andere keer");}

?>
```

Sla alles op in het bestand **oef6.php**, en download via FTP naar de server van je web-hoster. Open vervolgens het desbetreffende bestand:



De variabele \$prijs krijgt de waarde van 12. Zoals later zal blijken, kan dit een waarde zijn die bekomen is uit een database, of een waarde die is ingetypt door de bezoeker van je site.

If(\$Prijs <=10), kun je lezen als volgt: ‘Als de variable prijs kleiner is of gelijk aan 10...’

Indien bovenstaande klopt dan volgt de statement: echo ("Verkocht")

Let er vooral op dat na het if statement er een accolade volgt ( { } )!

Indien de statement tussen de ‘if’ haakjes klopt, dan wordt ALLES uitgevoerd die tussen de accolades staat (compound statement)

Indien het niet klopt wat er tussen ‘if’ haakjes staat, dan gaat de PHP code automatisch naar de ‘else’ statement.

Ook deze wordt gevolgd door ‘accolades’ ( { } )

In ons voorbeeld is de prijs NIET kleiner of gelijk aan 10, dus voert de code uit wat er bij ‘else’ staat.

In werkelijkheid kunnen mogelijks meer keuzes worden ingebouwd.

We kunnen voorbeeld stellen:

- als de prijs kleiner of gelijk is aan 10 dan kopen we
- als de prijs kleiner of gelijk is aan 12 dan denken we na
- in alle andere gevallen kopen we niet.

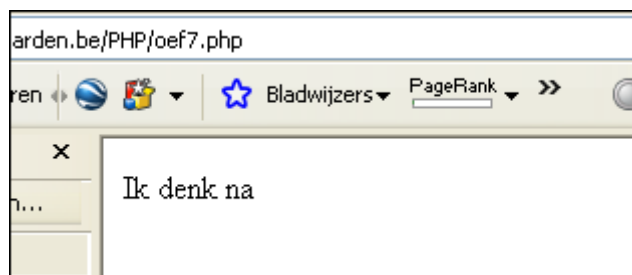
Open start.php en breng volgende code in de <body>.

```
<?php
$prijs=11;
if ($prijs<=10){
echo ("Verkocht");}
elseif ($prijs<=12){
echo ("Ik denk na");}
else
{echo ("Een andere keer");}

?>
```

We slaan dit bestand op als **oef7.php**. Via FTP laden we dit bestand naar de server.

Vervolgens openen we dit bestand in onze browser



Het ‘elseif’ statement kun je zo vaak als je wil na elkaar gebruiken.

## **6.2 Oefening ‘Huis kopen’**

Ik durf er om te wedden, dat je verlegen zit achter een echte oefening ☺.

Op basis van wat tot nu toe allemaal hebben doorgespit, zou je in staat moeten zijn om volgende oefening te maken.

Een huis kost 200.000 Euro.

Als iemand minder of gelijk aan 5000 euro tekort heeft, volgt de boodschap: Het contract komt eraan.

Als iemand minder of gelijk aan 50.000 euro tekort heeft, volgt de boodschap: With a little help of your friends

Als iemand minder of gelijk aan 75.000 euro tekort heeft, volgt de boodschap: Dringend opslag vragen

In alle andere gevallen, volgt de boodschap: Huur een caravan

In alle gevallen moet worden weergegeven hoeveel de bezoeker tekort komt!

Succes.

De oplossing vind je op [www.web-garden.be/cursus/huiskopen.pdf](http://www.web-garden.be/cursus/huiskopen.pdf)

## **6.3 Vergelijkingsoperatoren**

In bovenstaande voorbeelden hebben we de ‘vergelijkingsoperator’  $\leq$  (‘kleiner of gelijk aan’) gebruikt.

Zo bestaan er meerdere vergelijkingsoperatoren:

$=$  : is gelijk aan

$\geq$  : is groter of gelijk aan

$<$  : is kleiner dan

$>$  : is groter

$\neq$  : is niet gelijk aan

Let voornamelijk op de eerste:  $=$  . Voor de statement ‘is gelijk aan’, hebben we vermoedelijk de neiging om het ‘ $=$ ’ teken te gebruiken.

## **6.4 While ( )**

Bij het gebruik van de ‘while ( )’ statement bezorgt een lus-structuur. ‘While’ kunnen we vertalen als ‘terwijl’. Of iets vrijer: ‘zolang’.

Zolang een variabele een bepaalde waarde heeft, voeren we iets uit.

We hebben allemaal, op de schoolbanken, de ‘rekenkundige ‘ tafels op gedreund .

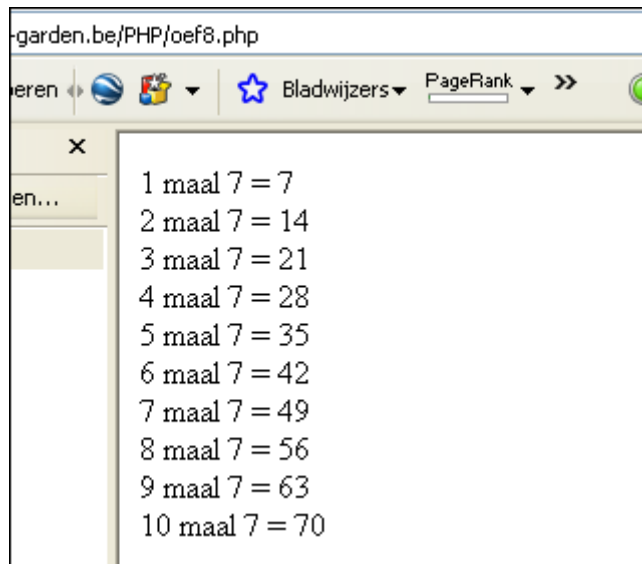
We maken nu even de tafel van 7 d.m.v. PHP.

Open start.php, en voeg onderstaande code toe in de <body>.

Sla alles op als **oef8.php**. M.b.v. FTP landen we dit bestand op in de map PHP op de server.

```
<?php
$steller=1;
$getal=7;
while($steller<=10) {
echo ("$steller.maal.$getal = " . ($steller*$getal). "<br>");
$steller++;}
?>
```

We openen het bestand op de server dmv onze internet-browser.



## 6.5 do...while ( )

Dit kunnen we als volgt weergeven:

*do {statement die wordt uitgevoerd tot de voorwaarde niet meer geldt }  
while (voorwaarde).*

Dit is meer dan een doordenkertje. Op het eerste zicht is er geen verschil met ‘while ( )’.  
Toch wel.

Bij while ( ) beginnen we met het vermelden van de voorwaarde.

In het voorbeeld in paragraaf 6.4 vermelden we eerst ‘while (\$steller<=10), daarna de uit te voeren actie.

Bij do...while, voeren we eerst de actie uit, en gaan dan na als de voorwaarde van 'while(\$teller<=10) nog geldt.

Dus bij do...while zullen we ALTIJD ten minste een keer een actie hebben.

Bij while( ) kan het zijn dat er helemaal geen actie wordt uitgevoerd (namelijk als \$teller > 10)

## **6.6 For ( )**

Ook bij het gebruik van het 'for ( ) ' statement maken we een lus structuur.

Algemeen kunnen we het gebruik van het 'for ( ) ' statement als volgt weergeven:

```
For (expressie1; expressie2; expressie3 )  
{ geneste statement(s) }
```

Laten we direct een praktisch voorbeeld geven :

```
<?php  
for ($k=1 ; $k<=5; $k++)  
{ echo (" $k <br>"); }  
?>
```

Bij het 'lezen' van een 'for' statement wordt 'expressie1' slechts één keer uitgevoerd...namelijk bij het openen (starten) van de lus.

In ons voorbeeld wordt krijgt de variabele \$k de waarde 1.

Vervolgens hebben we 'expressie2'. Dat is de voorwaarde voor het verder zetten van de lus.

In ons voorbeeld wordt nagegaan of de variabele \$k kleiner of gelijk is aan 5.

In geval van TRUE, worden de 'geneste' statements uitgevoerd (-statements tussen de accolades)

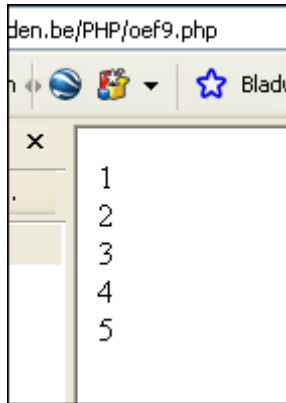
In ons voorbeeld wordt de waarde van de variabele \$k weergegeven op het scherm (echo).

Na het uitvoeren van de geneste statement, wordt de derde expressie ('expressie3') uitgevoerd.

In ons voorbeeld wordt de variabele \$k met 1 verhoogd ('\$k++')

Daarna wordt de 'expressie2' weerom gecontroleerd. Ingeval van TRUE, wordt de geneste statement terug uitgevoerd.

DE weergave van bovenstaande code in de browser ziet er als volgt uit:

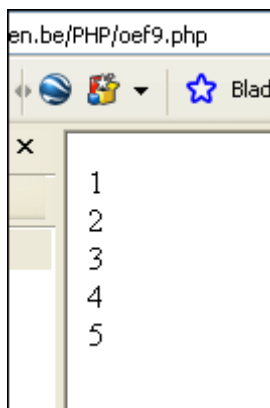


## 6.7 Break

Break is een statement om een lus te beëindigen. Na *break*; gaat het script door vanaf het eerste statement dat volgt op de sluit-accolade.

```
<?php
for($k=1; ; $k++) {
echo ("$k<br>");
if ($k= = 5)
break;
}
?>
```

Deze code heeft volgend resultaat



Interessant aan dit voorbeeld is dat we in het 'for' statement geen drie expressies meer hebben vermeld (slechts 2 : \$k=1 en \$k++).

De voorwaardelijke expressie wordt nu opgegeven, door een 'if' statement in de geneste statements .

Het weglaten van 'expressie2' wordt vervangen door een dubbel aanhalingsteken (; ;)

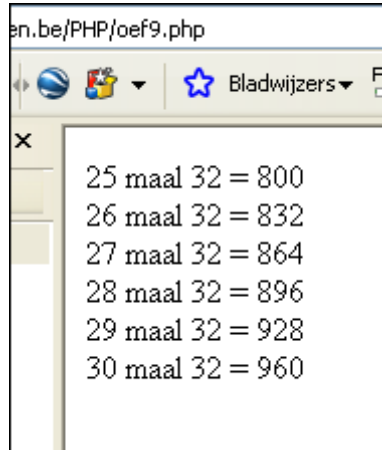
## 6.8 Oefening ‘tafel van vermenigvuldiging’

Even genoeg saaie (maar zo noodzakelijke ☺)definities.  
Weer tijd voor een oefening.

We gaan de tafel van 32 maken.

Maar alleen de bewerkingen tussen 25 en 30 mogen worden weergegeven op het scherm.

Dit moet u op het scherm krijgen☺:



De oplossing kunt U downloaden op <http://www.web-garden.be/cursus/tafelvan32.pdf>