

Hoofdstuk 5 : Variabelen en gegevenstypes in PHP

A. Variabelen

A.1 Definitie

Een variabele is een naam voor een bepaald gegeven of een waarde. De waarde van een variabele kan veranderen, terwijl de naam dezelfde blijft.

Voorbeeld: \$variabele = 0 \$variabele = "tekst"

De naam van een variabele begint met een '\$' teken. Voorbeeld: \$variabele = 2

Na het '\$' teken kan een underscore-teken ('_') staan of een letter. GEEN cijfer!!

Verder mag in de naam wel een cijfer gebruikt worden. Voorbeeld: \$variabele1 = 7

Zoals reeds eerder aangegeven is PHP hoofdlettergevoelig.

Dus: \$variabele is NIET hetzelfde als \$Variabele .

Namen van variabelen mogen geen spaties bevatten.

Een variabele heeft altijd de waarde die er als laatst is aan toegekend.

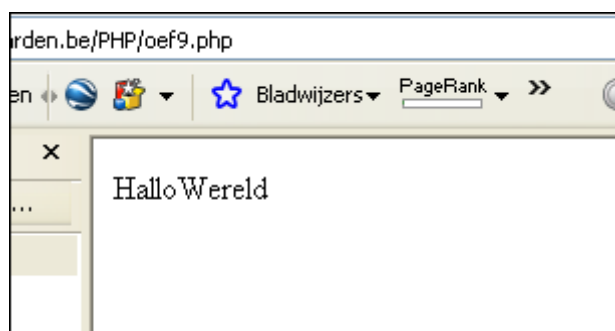
De waarde van een variabele is beperkt tot de pagina waarin ze gemaakt is.

A.2 Samenvoegen van variabelen.

Hierbij geven we aan hoe je variabelen kunt samenvoegen.

```
<?php
$var1 = "Hallo";
$var2 = "Wereld";
$var3 = $var1.$var2;
echo ("$var3");
?>
```

Bovenstaande code heeft volgend resultaat in je browser:



Het punt '.' teken voegt de twee variabelen samen tot 'Hallo Wereld'

We proberen nu het volgende:

```
<?php
$var1= "Hallo";
$var2= "Wereld";
$var3= $var1." ".$var2;
echo ("$var3");
?>
```

Bovenstaande code heeft volgend resultaat in je browser:



Door het toevoegen van het dubbele aanhalingsteken tussen \$var1 en \$var2, creëren we een extra spatie tussen Hallo en Wereld.

A.3 Samengestelde operatoren

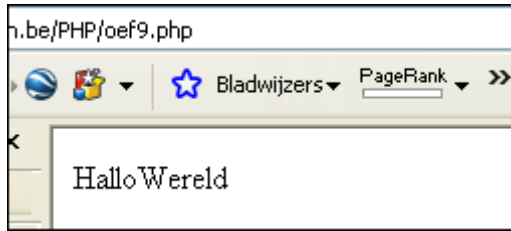
Een aantal bewerkingen kunnen korter worden weergegeven

| Traditioneel | Samengestelde operator |
|-------------------------|------------------------------------------------------------|
| \$var = \$var + 1; | \$var++; //tel 1 op bij \$var |
| \$var = \$var - 1; | \$var--; //trek 1 af van \$var |
| \$var = \$var + 10; | \$var+=10; //tel 10 op bij \$var |
| \$var = \$var - 10; | \$var-=10; //trek 10 af van \$var |
| \$var = \$var * 10; | \$var*=10; //vermenigvuldig \$var met 10 |
| \$var = \$var / 10; | \$var/=10; //deel \$var door 10 |
| \$var = \$var."Wereld"; | \$var.= "Wereld" ;// voeg de string "Wereld" toe aan \$var |

Laat ons de laatste samengestelde variabele eens oefenen:

```
<?php
$var1= "Hallo";
$var1.= "Wereld";
echo ("$var1");
?>
```

Bovenstaande code heeft nu volgend resultaat in je browser:



De tweede statement uit de code, lezen we als:

‘Voeg de string “Wereld” toe aan \$var1

B. Datatypes of Gegeventypes

Tot nu toe hebben we verschillende waarden toegekend aan een variabele. De ene keer was de waarde van een variabele een getal, de andere keer een stukje tekst.

De waarden toegekend aan een variabele kunnen bestaan uit verschillende gegevenstypes (datatypes).

In PHP onderscheiden we 7 gegevenstypes:

- Boolean
- Integer
- Double
- String
- Array
- Object
- NULL

B.1 Boolean

Gegevens van het type boolean kunnen 2 waarden hebben : true of false.

Aan elke waarde in PHP, is er steeds een boolean waarde gekoppeld.

Boolean waarden worden voornamelijk gebruikt in lussen of in vergelijkende statements.

Als voorbeeld van deze laatste, geven we hier: if()...else
(‘if’ kan worden vertaald als ‘als’).

```
<?php
$var=8;
if ($var < 9)
echo ("Peter is jarig");
else
echo ("Jan is afwezig");
?>
```

Bovenstaande code heeft volgend resultaat in je browser:

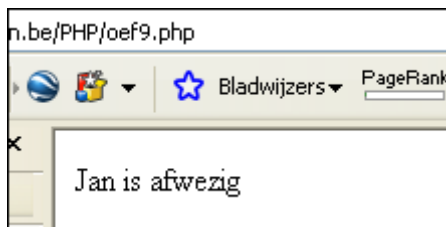


Als ('if') de waarde van variabele kleiner is dan 9 (true), geef dan de boodschap weer ('echo'): "Peter is jarig".

Als de waarde van \$var niet kleiner is dan 9 (false), geef dan de boodschap weer (echo): "Jan is afwezig"

We herhalen vorige oefening, maar geven \$var de waarde 10 (\$var = 10)

Dan krijg je dit in je browser te zien:



PS: Let er even op dat na 'if(\$var)' en na 'else', geen aanhalingstekens (;) wordt geplaatst.

B.2 Integers

Integers zijn gehele getallen (positief of negatief). Maw gegevens van het type 'integer' kunnen geen punten bevatten, of geen cijfers na de punt

Voorbeeld:

```
$var = 5;
```

PS: De gehele getallen kunnen decimaal, hexadecimaal of octaal zijn. Meer hierover vertellen lijkt me voor deze cursus te ver gezocht ☺

B.3 Doubles

Gegevens van het type 'doubles' zijn getallen met een punt , en cijfers na de punt

Voorbeeld:

```
$var = 5.0;  
$var1 = 34.567;
```

Men noemt dit ook wel floating-point getallen, of kort weg: floats.

Let er wel op dat we hier gebruik maken van een punt...*en niet van een komma!!*

B.4 Strings

Gegevens van het type 'strings', zijn tekenreeksen.

Voorbeeld:

```
$var = "Dit is een string";  
$var1 = "xytvw";  
$var2 = ""; // Dit is een lege string
```

Er zijn situaties in dewelke je de dubbele aanhalingstekens (") kunt vervangen door enkelvoudige aanhalingstekens (').

Ik raad echter aan steeds dubbele aanhalingstekens te gebruiken.

B.5 Array

B.5.1. Definities

Gegevens van het type 'array', is een groep variabelen met dezelfde naam, maar met verschillende waarden. De verschillende waarden worden bekomen door het gebruik van een 'index(notatie)'.

Voorbeeld:

Tot nu toe hadden we volgende variabelen zo weergegeven:

```
$kleur1 = "rood";  
$kleur2 = "geel";  
$kleur3 = "zwart";  
$kleur4 = "groen";
```

Als we gebruik maken van 'array', dan kunnen we bovenstaande variabelen als volgt weergeven:

```
$kleuren[0] = "rood"  
$kleuren[1] = "geel";  
$kleuren[2] = "zwart";  
$kleuren[3] = "groen";
```

We hebben hier dus een variabele met 1 naam (\$kleuren) die verschillen waarden kan hebben, bepaald door de index(notatie) ([0], [1], [2],.....)

De indexnotatie bij array's begint steeds met de waarde [0] !

Ik adviseer om geen enkele index over te slaan. M.a.w volgende reeks wordt ten stelligste afgeraden:

```
$kleuren[0] = "rood"  
$kleuren[1] = "geel";  
$kleuren[3] = "groen";
```

Afhankelijk van de PHP instellingen op de server van je web-hosting bedrijf, kan dit fout meldingen opleveren!

Voorbeeld:

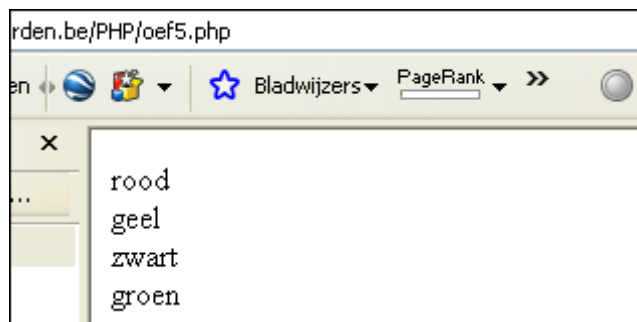
Open start.php, en voeg volgende code toe in de <body>:

```
<?php  
$kleuren[0] = "rood";  
$kleuren[1] = "geel";  
$kleuren[2] = "zwart";  
$kleuren[3] = "groen";  
for($i=0; $i<=3; $i++){  
echo("$kleuren[$i]<br>");}  
?>
```

Sla het bestand op in **oef5.php**, en download dit bestand via FTP naar de map PHP op de server.

Open het desbetreffende bestand nu via je browser (<http://www.mijnsite.com/PHP/oef5.php>)

Dit resultaat verschijnt in je browser:



B.5.2. De functie array()

Bovenstaand voorbeeld kunnen we ook als volgt weergeven:

```
$kleuren = array("rood", "geel", "zwart", "groen");
```

Ook in dit geval heeft \$kleuren[0] de waarde 'rood'.

Tot nu toe is de indexering van een array alleen op basis van cijfers, waarbij steeds begonnen wordt met het getal '0'.

Daarnaast kunnen we ook op een andere manier een array functie definiëren:
\$kleuren = array("a"=> "rood" , "b"=> "geel" , "c"=>"zwart" , "d"=>"groen");

a,b,c en d worden hierbij *sleutels* of *keys* genoemd.

PS: Wat betreft het gebruik van indexering bij array's, moet je wel even notie nemen van het gebruik van spaties.

[0] is niet hetzelfde als [0]
"a" is niet hetzelfde als "a "

Maw let er op dat je niet ongewenst spaties maakt bij het indexeren van een array.

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

```
<?php
$kleuren=array("a"=>"rood" , "b"=>"groen" , "c"=>"geel" , "d"=>"zwart");
while(list($sleutel, $waarde) =each($kleuren)){
echo("sleutel: $sleutel , waarde: $waarde <br>");}
?>
```

Sla op in oef5B.php. en download dit bestand via FTP naar de map PHP op de server.

Open het desbetreffende bestand nu via je browser (<http://www.mijnsite.com/PHP/oef5B.php>)

Open het bestand in je browser.



We staan even stil bij dit stukje code.
In de eerste lijn wordt een array gedefinieerd.

```
$kleuren=array( "a"=>"rood" , "b"=>"groen" , "c"=>"geel" ,  
"d"=>"zwart" );
```

De sleutel "a" krijgt de waarde "rood". De sleutel "b" krijgt de waarde "groen". Enz...

Deze koppels worden bewaard in de variabele \$kleuren.

In de tweede lijn krijgen we het ‘while’ commando. Dit commando wordt ergens anders nog besproken.

Letterlijk vertaald betekent ‘while’ = ‘zolang’

```
(list($sleutel, $waarde) =each($kleuren))
```

Met het list() = each() commando wordt een ‘koppel’ gelezen uit de variable \$kleuren. Het eerste element uit ieder koppel (=de sleutel) wordt dmv list bewaard in \$sleutel. Het tweede element uit ieder koppel wordt bewaard in \$waarde.

Met het ‘while’ commando wordt een lus-structuur gemaakt die ‘zolang’ wordt doorlopen tot alle koppels zijn gelezen.

De besproken code:

```
<?php
$kleuren=array("a"=>"rood" , "b"=>"groen" , "c"=>"geel" ,
"d"=>"zwart");
while(list($sleutel, $waarde) =each($kleuren)){
echo("sleutel: $sleutel , waarde: $waarde <br>");}
?>
```

zullen we frequent gebruiken bij het uitlezen van database-records.

B.5.3 Arrays lezen

We hebben bij de indexering gezien dat je kunt gebruik maken van [0] of van "a".

Stel dat we in beide gevallen de inhoud van een array willen weergeven op het scherm. Dan gebruiken we onderstaande code:

```
echo ($kleuren[0]); //Resultaat: rood
```

```
echo ($kleuren["c"]); //Resultaat: zwart
```

Let op de aanhalingstekens in het tweede geval!

B.6 Object (of Class)

Een object is een verzameling variabelen en functies voor het verwerken van de variabelen. Het gebruiken van objects, noemt men ook ‘objectgeoriënteerd programmeren’.

Na het afwerken van deze basis cursus, is het mijn bedoeling een extra handleiding te schrijven aangaande ‘objectgeoriënteerd programmeren’

B.7 NULL

Het gegevenstype NULL wordt gebruikt om aan te duiden dat een variabele geen waarde heeft (dus ook niet 'nul'!!).

B.8 Extra

Nu we alle gegevenstypes hebben beschreven, wil ik nog even terug komen met wat extra info betreffende 'boolean' gegevenstype:

-Bij integers of doubles is de boolean waarde 'false' als de waarde van de variabele 'nul' (0) is. In alle andere gevallen is de waarde 'true'

-Bij strings is de waarde 'false' als de string leeg is (" "), anders is de waarde 'true'

-Bij arrays is de waarde 'false' indien de 'array' geen elementen heeft, anders is de waarde 'true'.