

Hoofdstuk 2 : Syntaxisregels in PHP

Alvorens we met de programmeertaal zelf van staret gaan, is het belangrijker dat je op de hoogte bent van een aantal afspraken.

Zoals reeds aangegeven is PHP een programmeertaal. Voor iedere taal gelden er afspraken.

PHP code wordt steeds opgenomen tussen volgende tags:

```
<?php  
... .  
...  
?>
```

a. Witruimte

Om de code straks goed leesbaar te houden, kun je gebruik maken van witruimtes. Deze karakters worden niet weergegeven in het uiteindelijk resultaat op het scherm.

De volgende toetsen maken witruimtes die niet worden weergegeven:

- spatie
- tab
- 'Enter' (regelovergang)

Voorbeeld:

```
$test=10+10
```

is even goed als:

```
$test = 10 + 10
```

of als:

```
$test = 10 +  
15
```

Wanneer U echte een nieuwe regel wil beginnen, gebruik dan het HTML symbool

Voorbeeld:

```
$test = "dit is een voorbeeld<br\>  
tekst"
```

b. Hoofdletters

PHP is hoofdletter gevoelig.

Voorbeeld:

`$test` is niet hetzelfde als `$Test`

Gebruik steeds kleine letters!! Dit om “onvindbare fouten” te vermijden.
Tevens is het zo dat vele editors pas optimaal werken als je kleine letters gebruikt.

c. (Bestands)namen

Vaak wil je langere namen gebruiken, om de code beter leesbaar te maken. Gebruik dan steeds een underscore (`_`) tussen de woorden.

Voorbeeld:

`$eerste_test`

d. Statements of expressies

Bekijk even onderstaande PHP code:

```
<?php
$nummer1 = 10 + 10;
$nummer2 = 20 + 20;
$nummer3 = 30 + 30;
$totaal = $nummer1 + $nummer2 + $nummer3;
echo ($totaal);
?>
```

De bezoeker zal op zijn scherm het getal 120 zien. (120 is de som van 10 + 10 + 20 + 20 + 30 + 30)

Dit stukje code bestaat uit 5 statements of expressies. Een statement of expressie is een programmeerregel met opdrachten die iets uitvoeren.

Elk statement wordt afgesloten met een puntkomma ‘ ; ’

e. voorrang regels

Hier frissen we wat wiskundige kennis op ☺

Bekijk even onderstaand statement:

```
$nummer1 = 10 + 10 * 2
```

De uitkomst van deze is 30.

De ‘vermenigvuldiging’ heeft voorrang op de ‘optelling’

Wat met dit stukje code:

```
$nummer1 = 10/5/2
```

De uitkomst weergegeven door de browser zal 1 zijn. Op het eerst zicht behoorlijk logisch. Het had ook 0.04 kunnen zijn. Namelijk als we de bewerking uitvoeren van rechts naar links. PHP leest dus steeds code van links naar rechts.

f. Compound statements

We hebben in paragraaf 'd' aangegeven wat een statement is.

Wanneer we willen dat een serie statements, die bij elkaar horen, wordt uitgevoerd...kunt U deze combineren in een 'compound statement'. Compound statements wordt ingesloten tussen accolades {...}.

g. Commentaar toevoegen aan je code

Het is ten zeerste aangewezen om voldoende commentaar in je code op te nemen. Als je weken of maanden later je code terug leest, zul je sneller begrijpen wat je toen hebt uitgespookt.

We onthouden twee methodes om dat te doen.

```
//Dit is een commentaar lijn die loopt tot het einde van de regel
```

Inderdaad, na het teken "//" kun je de rest van de regel vullen met commentaar. De commentaar kan dus niet doorlopen op de volgende regel

```
/*Commentaar zoals ook gebruikt in HTML.  
Dit kan over meerdere regels */
```

Een stukje commentaar wordt begonnen met '/*' en eindigt met '*/'. De commentaar kan wel gespreid worden over meerdere regels.