Cursus PHP voor beginners.

Door middel van voorbeelden zal ik jullie proberen wat wegwijs te maken in de programmeertaal PHP. Om met PHP te starten is een goede basiskennis nodig van HTML, daar de één niet zonder de andere kan. PHP werkt even goed onder windows als onder unix, maar ik baseer mij hier op het unix systeem, aangezien dit het systeem is waarmee ik werk. Ik laat de windows gebruikers echter niet in de kou staan, ik laat zien hoe je PHP onder windows aan de praat krijgt.

Onder unix gebruik ik de HTML-editor Quanta om mijn HTML pagna's aan te maken. Een gewone teksteditor kan hiervoor ook dienen, maar met een HTML editor is het eenvoudiger werken. Onder windows kun je natuuurlijk kladblok gebruiken, maar gebruik liever ook een HTML editor, het zal je veel tikwerk besparen. Gebruik vooral geen WYSIWYG editor zoals Frontpage, want dergelijke editors voegen een heleboel nutteloze code toe aan je HTML pagina's, die wij hier kunnen missen als kiespijn.

Bij de voorbeelden staat steeds de file met de source code. Als je klikt op de link zal je browser het script tonen. Als je nu in de navigatiebalk van je browser de letter "s" tikt na de extensie .php zal je browser de source code van het script tonen. Vb : als je op btw.php klikt toont je browser de uitvoer van het script, als je er dan btw.phps van maakt, zal hij de source code van het script tonen.

Inhoud

Les 1

Wie heeft PHP uitgevonden ?

PHP werd uitgevonden door de Canadees van Deense afkomst Rasmus Lerdorf. In de herfst van 1994 wou Rasmus ontdekken hoe het mogelijk zou zijn om zijn applicaties via het web te publiceren. Hij schreef rap enkele server commando's die toelieten tellers en een gastenboek online te plaatsen. De Personal Home Page Tools waren geboren.

Formulieren zijn één van de belangrijkste elementen op het web, en gedurende het jaar 1995 schreef Rasmus dan ook een Form Interpreter om formulieren te beheren via het web.

Hij publiceerde zijn development omgeving gratis op het web, en de webdevelopers gemeenschap begon zich te interesseren voor PHP/FI en een werkgroep rond PHP/FI werd algauw gevormd.

PHP is thans aan versie 4.2.3 toe en de ontwikkeling ervan wordt verder afgehandeld door andere personen dan Rasmus Lerdorf.

PHP staat nu voor **H**ypertext **P**re**p**rocessor.

Wat is PHP ?

PHP is een programmeertaal (script taal). PHP wordt direkt in de HTML pagina's geïntegereerd en wordt geïnterpreteerd door de webserver. Aangezien PHP in de eerste plaats werd ontworpen voor het web, kan het enkel iets nuttigs doen voor webapplicaties. PHP heeft geen overbodige bagage, wat wel het geval is voor enkele concurerende programmeertalen.

Hoe functioneert PHP ?

Zoals reeeds gezegd, PHP wordt geïnterpreteerd door de webserver. Wat gebeurt er nu juist :

De browser stuurt een HTTP verzoek naar de webserver (opvragen van een pagina). De webserver reageert op het verzoek, leest de PHP webpagina, voert het PHP script uit, en stuurt dan het document terug naar de browser.

Omdat PHP wordt uitgevoerd door de server, zal de client nooit de PHP code te zien krijgen, daar hij het uiteindelijke resultaat voorgeschoteld krijgt als een webpagina in HTML formaat.

Wat is het vershil tussen Java en PHP ?

Misschien kennen jullie reeds al wat Javascript. Javascript wordt rechtstreeks in de HTML code geïntegreerd. De client zal steeds de code te zien krijgen als klare tekst in zijn browser als hij de source van de pagina bekijkt.

Waarom ? Javascript wordt enkel geïnterpreteerd nadat je de pagina opgevraagd hebt. Javascript wordt uitgevoerd door de client en niet door de server. Enkele van de gekende minpunten hiervan zijn : wat javascript betreft heeft iedere browser zo zijn eigen eigenaardigheden, en voor hen die niet over javascript beschikken, niets werkt van de code.

De voordelen van PHP zijn evident. Als het werkt, werkt het in iedere browser. Enige voorwaarde is, dat je provider PHP ondersteund op hun servers waarnaartoe je je pagina's verstuurd.

Wat kost PHP ?

Helemaal niets ! PHP werd in open-source geplaatst, waardoor de broncode voor iedereen vrij beschikbaar is. Als je een Linux of andere unix distributie gebruikt, is de kans groot dat PHP al met je systeem meegeleverd is. De laatste versie kun je steeds downloaden op http://www.php.net.

PHP wordt over het algemeen tesamen gebruikt met de webserver apache, maar kan ook met het even welke andere webserver worden gebruikt, zolang die maar PHP ondersteund. Apache is de populairste webserver op het net, en bovendien geheel gratis. Ik gebruik hier dan ook Apache als webserver. Later meer over de installatie ervan.

PHP ondersteuning bij uw provider

PHP zoals Apache zijn gratis, dus je kunt die zonder problemen installeren op je computer thuis. Je kunt dan prachtige scripts ontwerpen maar als je dan je eigen werk met anderen wilt delen, zal je provider PHP moeten ondersteunen op zijn webserver of niemand zal kunnen meegenieten van je eigen werk. Voor de telenet gebruikers, telenet heeft geen PHP ondersteuning. Na een mailtje te hebben gestuurd om te vragen of er eventueel PHP ondersteuning was, kreeg ik als antwoord dat HTML voor dummies een goed boek was om te starten met HTML om mijn webpagina online

te plaatsen.

Ofwel draai je dan je eigen webserver of moet je op zoek gaan naar een webhost die PHP ondersteund. Eéntje die ik ken die PHP ondersteund en gratis is kun je vinden op http://www.free.fr.

Is PHP gemakkelijk aan te leren ?

Ja, PHP is zonder meer gemakkelijk aan te leren en is veel eenvoudiger dan JavaScript en Perl. Als men al ervaring heeft met een andere programmeertaal, zal PHP natuurlijk eenvoudiger te leren zijn. In het begin zal ik hier alles zo eenvoudig mogelijk proberen te houden, maar naargelang de lessen vorderen, zal het natuurlijk gecompliceerder worden.

Onthoud echter, al smedend wordt men smid of oefening baart kunst. Als je stap voor stap werkt, de voorbeelden goed nakijkt en je werkelijk geïnteresseerd bent in PHP, zal alles wel meevallen.

PHP versus Perl

Perl is ook een veelvuldig gebruikte programmeertaal op het web. Perl is in tegenstelling tot PHP enorm gecompliceerd. Perl werd ontworpen in het jaar 1986, een tijd waarop internet nog voor velen een droom leek. PHP zag het levenslicht 10 jaar later en werd dan ook rechtstreeks geschreven voor het web.

PHP versus ASP

ASP met VBScript of JScript is ook geen echt alternatief te noemen voor PHP, daar deze uitvinding van Microsoft enkel werkt met Internet Information Server, en deze webserver vertegenwoordigd maar een heel klein percentage van de webservers.

Wat kun je met PHP doen ?

PHP heeft zowat alles ter beschikking om volgende zaken te maken :

- Pagina tellers
- Een gastenboek
- Versturen en beheren van cookies
- Huidige datum en uur integeren in jouw pagina
- Controle of een webformulier volledig werd ingevuld
- Een webwinkel inrichten
- Een automatische Email service
- enz ....

Met deze kleine voorstelling van PHP wordt het tijd om de koe bij de horens te vatten en binnen te dringen in de wondere wereld van PHP.

Les 2 : Installatie van de Apache webserver en PHP

FreeBSD

```
root # cd /usr/ports/net/apache13

root # make && make install clean
```

Nadat apache is geinstalleerd :

```
root # cd /usr/ports/net/mod_php4

root # make && make install clean
```

Nadat je dit commando hebt gerund, krijg je een mogelijkheid om enkel opties te kiezen. Kies de volgende opties :

- GD2
- zlib
- imap
- MySQL
- FTP
- gettext

Voor diegenen die al gewoon zijn te werken met een database en PostgreSQL gebruiken, kunnen dan deze optie aanvinken i.p.v. MySQL of je kunt ze natuurlijk beiden aanvinken.

Na de compilatie beschik je dan over een apachewebserver met ondersteuning voor PHP.

Debian GNU/Linux

root # **apt-get install apache php4 php4-mysql php4-imap php4-gd mysql-client mysql-server**

Voor andere Linux distributies even zelf kijken hoe je dat allemaal moet installeren.

Windows

Onder windows heb je een gemakkelijke en moeilijke manier om PHP te installeren. Ik geef je hier de gemakkelijke. Surf naar www.easyphp.org en download daar easyphp voor windows. Enig minpuntje is dat het in het Frans is, download dan ook de engelstalige module. Installeren doe je dan gewoon door te dubbelklikken op de file die je hebt gedownload. Na de installatie krijg je in de systray een E met een flikkerend rood puntje dat aangeeft of apache en mysql actief is of niet.

Als je de moeilijke weg wilt bewandelen, ga naar www.php.net en download daar PHP, ga vervolgens naar www.apache.org en download daar de apache server voor windows. Voor verdere installatie, volg de help op de bewuste pagina's. Wens je ook MySQL, dan vindt je dat op www.mysql.org.

Als apache en PHP op jouw systeem zijn geïnstalleerd kunnen we beginnen met het serieuze werk. De configuratie van apache is geen onderwerp dat ik hier zal behandelen, er staan reeds howto's over op mijn site en wat windows betreft, met easyphp is alles van de eerste keer in orde, gewoon rechts klikken op de E en kiezen voor local, en je webbrowser zal opgestart worden en de index pagina tonen die te vinden is in c:\Program Files\EasyPHP\www en dat is dan ook de plaats waarin je je pagina's die je opmaakt door middel van PHP zult moeten plaatsen

Voor de FreeBSD en Linux gebruikers, normaal installeert je distributie alles in één keer, en is je httpd.conf ook aangepast. Zorg er wel voor dat je DocumentRoot verwijst naar bv /home/jouw_usernaam/homepage, want in de default installatie directory heeft alleen root schrijfrechten.

Om te controleren of PHP daadwerkelijk werkt op ons systeem, zullen we onze eerste pagina schrijven die we info.php noemen met volgende inhoud :

```
<html>
```

```
<head>
<title>PHP info</title>
</head>
<body>
<h1>Informatie over PHP</h1>
<?php
phpinfo();
?>
</body>
</html>
```

Sla deze pagina nu op in je DocumentRoot directory, start je browser en surf naar **http://localhost/info.php** of **http://127.0.0.1/info.php**. Als alles goed gaat moet je nu de informatie zien over de bij jouw geïnstalleerde PHP versie.

Om te vergroten, klik op de afbeelding

Mocht je deze informatie niet te zien krijgen, dan is er iets mis met de configuratie van apache, volg de aanwijzingen van je distributie, zodat apache op de juiste wijze wordt geconfigureerd.

Les 3 : Eerste praktische stappen in PHP

Laten we beginnen met PHP. In deze les zal ik jullie vertrouwd laten worden met de PHP notatie, hoe op een eenvodige wijze gegevens te publiceren, variabelen en arrays.

De PHP notatie

De PHP code wordt rechtstreeks in de HTML pagina ingevoerd. Waar de code wordt ingevoerd speelt geen enkele rol. Het voornaamste is dat je aangeeft waar de code begint door middel van de tag **<?php** en waar de code eindigt door middel van de tag **?>**

Deze tags zijn de lange versie, conform de XML richtlijnen. Je kunt de tags ook schrijven op de korte manier als **<?** en **?>** en men heeft ook gedacht aan de liefhebbers van ASP **<%** en **%>**

Ik zal hier steeds gebruik maken van de lange versie, en jullie worden verzocht om het ook zo te houden om geen verassingen te krijgen bij het aanmaken van jullie scripts. Vanaf de versie 4.2.2 van PHP is het gebruik van de korte tags (short tags) bij default op off geplaatst in het php-ini bestand alsook de register_globals. Al mijn scripts worden geschreven met de standaard instellingen van php-ini vanaf versie 4.2.3 en die eveneens werken met de lagere versies. Dat deze instellingen bij default van de ene versie tot de andere verschillen, heeft te maken met de veiligheid van je

scripts, en wie zijn wij om te twijfelen daaromtrent. De makers van PHP zullen het wellicht beter weten :-)

Ons eerste script : Hello World !

Zoals in elke programmeertaal is tonen van Hello World ! het eerste wat geleerd wordt, en ik zal deze traditie niet breken :-)

Gegevens tonen in een HTML pagina wordt verkregen door het commando *echo*.
Maak nu een nieuwe lege HTML pagina aan.
Daarmee bedoel ik voor in het vervolg van de lessen, een pagina bevattende :

```
<html>
<head>
<title></title>
</head>
<body>
</body>
</html>
```

Vul nu in tussen de title tags :

Hello World voorbeeld

En tussen de body tags :

```
<h1>Het wereldberoemde Hello World script</h1>

<p>
<?php
echo "Hello World !";
?>
</p>
```

En sla de pagina op als hello.php.

Als je nu surft naar **http://localhost/hello.php** zul je begroet worden door het Hello World script.

Wil je nu dat Hello World ! vet wordt afgedrukt, geen probleem, verander het echo commando door **echo "Hello World !";**

Zoals je ziet kunnen typische HTML tags en PHP instructies moeiteloos door elkaar gebruikt worden.

Een punt-comma op het einde van de regel

Je vraagt je misschien af, waarom een ; op het einde van de regel. Wel, de ; is onontbeerlijk op het einde van de lijn. Net zoals onder C++, Java of JavaScript, een PHP code wordt beëindigt met een ; Indien je dit vergeet en je code bevat meerdere lijnen, zullen de errors je om de oren vliegen. Natuurlijk bevestigen de uitzonderingen de regel en deze uitzonderingen zijn :

functies gedefinieerd door het codewoord function() alsook voor if, else, while, switch enz.. Daar is het dan verboden een ; te plaatsen, maar alles op zijn tijd als we zover zijn.

Zoals je ook merkt, bij het commando echo, wordt hetgeen dat moet worden getoond voorafgegaan door aanhalingstekens en afgesloten door aanhalingstekens. Je kunt dan ook geen aanhalingstekens meer gebruiken tussen de eerste en laatste aanhalingstekens. Dit brengt natuurlijk al een probleem met zich mee als je bijvoorbeeld de tekst HELLO WORLD !

gecentreerd wilt tonen op de HTML pagina. De code wordt dan

echo " Hello World ! ";

Als je dit nu opslaat in hello.php en dan de pagina opvraagt, zul je een parse error krijgen. Zoals :

Parse error: parse error, expecting `',' or `';'' in /
home/serge/homepage/hello.php on line 8

Nu om aanhalingstekens te gebruiken binnen de eerste en laatste aanhalingstekens, moeten wij die *escapen* zoals dat gebruikelijk is in iedere programmeertaal. Het *escapen* gebeurt door middel van de backslash "\". Met de backslash vertellen wij PHP dat hij het letterteken na de backslash letterlijk moet opvatten en deze niet moet interpreteren. We zullen die backslash nog veelvuldig moeten gebruiken, niet alleen om aanhalingstekens te *escapen*, maar alles op zijn tijd. Onthoud als ik spreek over *escapen*, dat ik het over die fameuze backslash heb.

Hoe moet die echo regel er dan uiteindelijk uitzien :

echo "<div align=\"center\">Hello World ! ";

En op die wijze zal de tekst Hello World ! correct gecentreerd worden weergegeven in je pagina.

Om dit gemakkelijk te onthouden, schrijf je regel zoals je gewoon bent, kijk of de regel begint met een aanhalingsteken en eindigt met een aanhalingsteken. Indien ja, plaats dan een backslash voor elk aanhalingsteken tussen deze eerste en laatste aanhalingstekens.

Commentaar

Een gouden raad : gebruik commentaar in je scripts. Specifieer door middel van commentaar wat elke lijn van je code uitvoert. Dit klinkt natuurlijk belachelijk bij korte scripts, maar eenmaal je grotere scripts gaat aanmaken en je moet een paar maanden later wat aan je script aanpassen, zul je zeker vergeten zijn welke rol deze lijn heeft of wat de betekenis kan zijn van deze tiende sluitende accolade. Wedden ?

Syntax van commentaar onder PHP

Eerst en vooral, commentaar beïnvloed de werking van het script niet. Het is wat het moet zijn, gewoon commentaar. Commentaar kan men op de volgende wijze invoeren :

```
//Ziehier een korte commentaar regel
```

of

```
/* Dit commentaar kan gespreid worden
over meerdere regels */
```

of

```
# Dit is commentaar die we
# gewoon zijn onder Perl
# of shell scripting.
```

Ter herinnering : als je commentaar wilt plaatsen in de HTML syntax :

<!-- en het commentaar wordt afgesloten met -->

of

```
 //-->
```
Commentaar kan ook nuttig zijn om fouten op te sporen. Je hebt enkele lijnen toegevoegd aan je code, en plots werkt je script niet meer ? Je kunt dan één lijn uit commentariëren, vervolgens een andere om zo de fouten te vinden. Een ander gebruik is, om een bestaande lijn code te commentariëren en die te vervangen door een nieuwe. Als er iets fout gaat, heb je nog steeds de originele lijn code ter beschikking om die eventueel terug te plaatsen.

Unix gebruikers weten dit, alhoewel het vaak niet gedaan wordt, maar als je config bestanden moet veranderen op jouw systeem, plaats je dan de bestaande regel in commentaar en voeg je de nieuwe toe die je wilt hebben. Bij belangrijke config bestanden kan dit je systeem om zeep helpen zodat je moet overgaan tot herinstallatie in het uiterste geval. Commentaar kan dus werkelijk belangrijk zijn.

De variabelen

Geen enkele programmeertaal kan zonder variabelen, en PHP maakt hierop geen uitzondering.

Wat is een variabel ?

Variabelen zijn plaatsen in het geheugen van de computer waar men bepaalde waarden kan bewaren. Deze waarden blijven dan beschikbaar gedurende de ganse uitvoering van een programma. Tijdens het uitvoeren van een script kun je deze plaatsen gebruiken om gegevens in te bewaren, aan te passen en nieuwe waarden toe te kennen, naargelang de behoefte van het script. De levensduur van deze waarden eindigt met het afsluiten van het programma of script. Een waarde kan een getal (*integer*) zijn, een tekenreeks (*string*), een waarheids test (*boolean*), een geheel getal van 0 tot 255 (*byte*), een getal met zwevende komma (*float*), *Double, Date, Currency, Variant* enz ...

In het begin moet elke variabel gedefinieerd worden in de meeste programmeertalen. Als je dergelijke programmeertaal aanleert, brengt dat al serieus wat stress mee van in het begin. Voor elke variabel die je gebruikt zonder deze vooraf te hebben gedefinieerd, krijg je een error.

Onder PHP geen stress, PHP werd in de eerste plaats geschreven voor het web, en werd ontdaan van alle niet nodige onderdelen van een klassieke programmeertaal. Onder PHP moet je je hoofd niet breken over welk type variabel het nu juist gaat, je gebruikt variabelen "on the fly" en PHP zal zelf wel uitmaken over welk type variabel het gaat. Onder PHP moet je geen variabelen op voorhand definiëren.

Onder PHP **MOET** elke variabele beginnen met het "$" teken. Vermijd accenten, spaties en namen van variabelen die eigen zijn aan de programmeertaal. Variabelen zijn hoofdletter gevoelig, $naam en $Naam of $NaAm zijn drie verschillende variabelen, die elk een andere waarde kunnen hebben. Variabelen een naam geven specifiek aan de programmeertaal mag niet, voorbeeld : $if, $while, $else enz ...

Een underscore "_" is toegelaten en dient in vele gevallen om een verboden ruimte in te vullen. voorbeeld $naam_serge. Maar genoeg theorie nu, wat zou je denken van een praktisch voorbeeld ?

Maak een nieuw HTML document aan met de naam variabel1.php. Plaats nu tussen de en tags de volgende code :

```
<?php
$naam="Essetee";
echo "Bedankt <strong>$naam</strong> voor deze cursus !";
?>
```

Surf nu naar **http://localhost/variabel1.php** en de tekst **Bedankt Essetee voor deze cursus !** zou getoond moeten worden.

Wat doen deze twee lijnen code ?

We initialiseren de string Essetee aan de variabel $naam. De tweede lijn print de inhoud van de variabel af in een wat meer interessante omgeving. Onthou, de variabel staat links van de waarde die je eraan wilt toekennen. Zoals je merkt, wordt de inhoud die we willen toekennen aan de variabele omringd door aanhalingsteken. Normaal is het verplicht om zo te werken met strings. Wil je een getal toekennen aan een variabele, moet deze niet tussen aanhalingstekens worden geplaatst. Nu mocht je vergeten die aanhalingstekens te plaatsen als je een string wilt definieren als variabele, dan zal PHP daar niet moeilijk over doen en proberen er het beste van te maken, wat dan ook tot vreemde resultaten kan leiden.

voorbeeld :

```
<?php
        $naam=serge;
        $getal=1;
        $totaal=$naam + $getal;
        echo "het getal = $totaal";
?>
```

PHP zal hier geen fout geven en leuk meedelen dat de waarde van het getal **1** is. Waarom ? Wel heel eenvoudig, $naam bevat zogezegd het getal serge, wat PHP niet kent, en hij geeft dan zelf de waarde 0 aan de variabele $naam.

Ander voorbeeld :

```
<?php
        $naam="serge";
        $getal=1;
        $totaal=$naam + $getal;
        echo "het getal = $totaal";
?>
```

Dit geeft opnieuw hetzelfde resultaat, het getal heeft de waarde 1. Je probeert hier een string op te tellen met een geheel getal. Opnieuw zal PHP de string als het getal 0 initialiseren.

Nog een voorbeeld :

```
<?php
        $naam="serge";
        $getal="1";
        $totaal=$naam + $getal;
        echo "het getal = $totaal";
?>
```

Dit geeft opnieuw het getal 1 als resultaat. PHP probeert hier een waarde toe te kennen aan de strings, wat lukt voor $getal, want dit is gelijk aan 1, voor $naam vindt hij geen waarde en geeft terug de waarde 0.

Zo zie je maar dat het vergeten van aanhalingstekens niet tot fouten leidt, maar dat de uitkomst soms niet is wat je zou verwachten. Ziehier een voorbeeld dat wel correct werkt met of zonder aanhalingstekens :

```
<?php
        $naam="12";
        $getal="1";
        $totaal=$naam + $getal;
        echo "het getal = $totaal";
?>
```

Nu zal het totaal wel correct worden getoond, namelijk 13. Wat als we nu 12 vervangen door 1.2, wel dan zal de uitkomst 2.2 worden. Plaats je echter de waarde 1,2 (een komma ipv een punt), dan wordt de waarde van het getal 2.

Waarom ? Wij zijn gewoon om onze decimalen aan te geven door middel van een komma, in een programmeertaal worden die aangeduid door middel van een punt. (Remember the good days of cobol, komma is decimal point) Opnieuw probeert PHP er het beste van te maken, hij gaat de string na, en het eerste wat hij vindt dat op een getal lijkt is de 1 en de komma en de rest negeert hij gewoon, en 1 + 1 is nog steeds twee, ook onder PHP :-)

Wat doet het volgende voorbeeld ?

```
<?php
        $naam=1,2;
        $getal=1;
        $totaal=$naam + $getal;
        echo "het getal = $totaal";
?>
```

Yep, je hebt prijs. Eindelijk een parse error. Aangezien 1,2 geen getal is voor PHP kun je dus ook die waarde niet toekennen aan de variabele $naam en geeft PHP dan dienovereenkomstig een foutmelding.

Voor beginners zijn dat veel voorkomende fouten. Als je echter onthoudt dat strings tussen aanhalingstekens worden geplaatst en dat getallen zonder aanhalingstekens worden gedefinieerd, dan is er niets aan de hand. Als je al eens vergeet een punt te plaatsen voor een decimaal getal, zal PHP stoppen met het script, een parse error geven en het nummer van de lijn vermelden waar de fout staat. Zoniet, kan het heel moeilijk worden bij grotere scripts om de fout te vinden als de uitkomst iets totaal anders is dan je had verwacht.

Laten we nu terug eens onze code bekijken die we aanvankelijk gebruikten :

```
<?php
$naam="Essetee";
echo "Bedankt <strong>$naam</strong> voor deze cursus !";
?>
```

Voor iemand die al een programmeertaal kent, is dit niet logisch. De correcte schrijfwijze zou moeten zijn :

```
echo "Bedankt " + $naam + " voor deze cursus !";
```

Wel onder PHP is dit niet nodig, en kan dus bijgevolg ook geen verwarring zaaien met het + teken bedoeld om iets op te tellen, zoals dit het geval is bv onder JavaScript. Nu op iedere regel is een uitzondering. Als je waarden recupereert uit functies of arrays gaat die vlieger niet meer op.

De concatenation operator Punt (.)

Als je meerdere strings op dezelfde lijn wenst weer te geven dan gebruik je de operator punt (.). Je kunt die natuurlijk altijd gebruiken, niemand belet het jou, maar het zal verplicht zijn deze te gebruiken als de waarden die je wilt afprinten afkomstig zijn uit een functie of array. We nemen nu terug ons voorbeeld en passen dit wat aan zodat we met de operator punt kunnen werken :

```
echo "Bedankt " . $naam  . " voor deze cursus !";
```

Dit zal exact hetzelfde afprinten. Zoals je ziet plaats je geen aanhalingstekens rond de variabele $naam. Wat indien je die toch tussen aanhalingstekens plaatst ? Wel het resultaat blijft hetzelfde. Dat we er geen aanhalingstekens rond plaatsen heeft zijn bedoeling waar we later nog op zullen terugkomen. Ik zal dan wel op dat moment aangeven wat er fout zal gaan als we eenmaal zover zijn.

De HTML break of een einde regel teken ?

De break
 wordt veel gebruikt in HTML pagina's. De break zorgt ervoor om naar een nieuwe lijn over te gaan, let wel, in het venster van de browser. PHP kent ook zoiets, namelijk een new line. Om deze twee te vergelijken nemen we terug ons variabel1.php voorbeeld, dat we wat gaan aanpassen :

```
<?php
$naam="Essetee";
echo "Bedankt <strong>$naam</strong> voor deze cursus !";
?>
```

En we veranderen dit in :

voorbeeld : variabel2.php

```
<?php
$naam="Essetee";
echo "Bedankt <strong>$naam</strong> voor deze cursus !<br>";
echo "Ik hoop dat ik er iets van kan leren";
?>
```

Dit wordt dan getoond in je browser als :

```
Bedankt Essetee voor deze cursus !
Ik hoop dat ik er iets van kan leren
```

Dit wordt dus correct weergegeven. De
 die we hebben toegevoegd doet zijn werk goed in de browser. Maar wat als je nu de source van de pagina bekijkt ? Wel dat geeft dan :

```
<html>
<head>
<title>variabel1</title>
</head>
<body>
<p>
Bedankt Essetee voor deze cursus !<br>Ik hoop dat ik er iets van kan
leren</p>
</body>
</html>
```

Zoals je ziet, geen enkel nieuwe lijn werd toegepast op de achtergrond. Als je vaak het commando echo gaat gebruiken, zal een lijn zonder onderbreking worden gevormd op de achtergrond. Als er iets fout gaat en je wilt even de pagina bron bekijken, wordt dit dan knap lastig. PHP heeft daarin iets voorzien, het commando new line \n.

We gaan dit even toepassen op ons script :

voorbeeld : variabel3.php

```
<?php
$naam="Essetee";
echo "Bedankt <strong>$naam</strong> voor deze cursus !<br>\n";
echo "Ik hoop dat ik er iets van kan leren";
?>
```

Merk terug de bakslash die we hier gebruiken voor de letter n, laat geen plaats tussen de backslash en de n. We escapen hier de n en daarmee geven we te kennen aan PHP dat hij de n niet moet zien als een letter dat hij moet afprinten, maar wel degelijk als een nieuwe lijn teken. De pagina zal nu in je browser op exact dezelfde wijze worden weergegeven, maar als je nu de broncode van de pagina bekijkt geeft dit dan het volgende :

```
<html>
```

```
<head>
<title>variabel1</title>
</head>
<body>
<p>
Bedankt Essetee voor deze cursus !<br>
Ik hoop dat ik er iets van kan leren</p>
</body>
</html>
```

Zoals je ziet is er nu ook een nieuwe lijn op de achtergrond. De volgende tekens kun je gebruiken :

Het escape teken	\n	\r	\t
Uitleg	Nieuwe lijn	Nieuwe paragraaf	Maakt een tab aan

Ik wil nu echter dan mijn naam tussen aanhalingstekens wordt geplaatst als ik de pagina afprint. Zoals reeds gezegd, aanhalingstekens duiden het begin en het einde aan van een karakterreeks. Hetzelfde voor als je een prijs wilt uitdrukken in dollars of indien je een backslash wilt laten afprinten. Wel al deze tekens moeten we escapen.

Voorbeeld :variabel4.php:

```
<html>
<head>
<title>variabel1</title>
</head>
<body>
<p>
<?php
$naam="Essetee";
echo "Bedankt \"Essetee\" voor deze cursus !<br>\n";
echo "Of zo wordt dit ook correct geschreven<br>\n";
echo "Bedankt \"$naam\" voor deze cursus !<br>\n";
echo "Ik hoop dat ik er iets van kan leren<br>\n";
echo "Ik vind het fijn dat PHP \$0 kost<br>\n";
echo "Ik begin stillaan wel iets te begrijpen over het escape teken
\\<br>\n";
?>
</p>
</body>
</html>
```

De arrays

Variabelen zijn mooi, maar wat als we nu meerdere waarden willen toekennen aan dezelfde variabele ? Wel in dit geval gebruiken we de arrays. Arrays worden ook wel lijsten genoemd. In tegenstelling tot andere talen, hoeven arrays niet op voorhand gedimensioneerd te worden, we maken arrays aan "on the fly" wanneer we er eentje nodig hebben. PHP kent associatieve arrays, wat totaal onbekend is in JavaScript en werkelijk uitzonderlijk is, maar daar komen we nog later op terug. Arrays hebben een korte notatie en een lange notatie. Hier gaan we dan.

Arrays in de lange notatie

Wat kunnen we nu in een array plaatsen ? Laten we eens beginnen met de

dagen van de week. Per default, is de waarde van de index van de eerste array steeds 0, dus de computer begint te tellen vanaf 0 en niet zoals wij gewoon zijn vanaf 1. Niet vergeten, we tonen dit aan de hand van een voorbeeld :

Voorbeeld : dagen1;php

```
<html>
<head>
<title></title>
</head>
<body>
<p>
<?php
$dag[0]="Zondag";
$dag[1]="Maandag";
$dag[2]="Disndag";
$dag[3]="Woensdag";
$dag[4]="Donderdag";
$dag[5]="Vrijdag";
$dag[6]="Zaterdag";

// Alleen woensdag tonen

echo $dag[3];
?>
</p>
</body>
</html>
```

Als je dat nu in je browser bekijkt zal hij "Woensdag" tonen. Een week begint op zondag en eindigt op zaterdag. Dus is woensdag eigenlijk de vierde dag van de week, maar aangezien de computer van 0 telt, moeten we hier de waarde 3 toekennen, wat in feitelijk staat voor 4 als je begint te tellen vanaf 0.

Arrays in de korte notatie

We gaan nu terug de dagen van de week opnemen in een array, maar dit doen we ditmaal zo :

voorbeeld : dagen2.php

```
<html>
<head>
<title></title>
</head>
<body>
<p>
<?php
$dag=array
("Zondag","Maandag","Dinsdag","Woensdag","Donderdag","Vrijdag","Zaterdag");

// De zondag afprinten

echo $dag[0];

?>
</p>
</body>
</html>
```

Als je dit nu in je browser ziet, print hij nu Zondag af. Dit is de schrijfwijze van arrays die ik steeds zal gebruiken, en wat mij betreft, wel de eenvoudigste is om te gebruiken. Zoals je ziet moeten in de array alle velden tussen aanhalingstekens staan en moet ieder veld gescheiden zijn door middel van een komma.

Allemaal goed en wel, maar wat kunnen we nu doen met die array. Laten we even voorstellen dat je de bezoekers op je pagina wenst mede te delen welke dag het vandaag is. Wel met een array en een specifieke functie van PHP is dat perfect mogelijk. Die functie heet date. Ik zal nu even demonstreren hoe we dat doen, maar op de functie date komen we later nog uitgebreid op terug.

voorbeeld : dagen3.php

```
<html>
<head>
<title>De dag van de week bepalen</title>
</head>
<body>
<p>
<?php

// bepalen welke dag we zijn vandaag
$vandaag=date("w");

$dag=array
("Zondag","Maandag","Dinsdag","Woensdag","Donderdag","Vrijdag","Zaterdag");

echo " Vandaag zijn we $dag[$vandaag]<br>\n";
?>
</p>
</body>
</html>
```

Als je dit toont in je browser, moet hij de dag opgeven die we vandaag zijn. Zoals je ziet maken we gebruik van de variabele $vandaag om te weten welke dag we zijn. De functie date("w") geeft hier een getal terug, dat overeenstemt met de dag van de week, beginnende de Zondag met als waarde 0. Dus in plaats van nu zelf een getal in te vullen tussen de haakjes, gebruiken we hier de waarde $vandaag.

Nu weten we dat de array $dag 7 elementen bevat. Maar we zullen dikwijls arrays dynamisch aanmaken, zonder op voorhand te weten hoeveel elementen die array zal bevatten. Wil je nu weten hoeveel elemeten een array bevat, dan heb je de fukntie count() ter beschikking.

voorbeeld : count.php

```
<html>
<head>
<title>De dag van de week bepalen</title>
</head>
<body>
<p>
<?php

// bepalen welke dag we zijn vandaag
$vandaag=date("w");
```

```
$dag=array
("Zondag","Maandag","Dinsdag","Woensdag","Donderdag","Vrijdag","Zaterdag");

echo " Vandaag zijn we $dag[$vandaag]<br><br>\n";

$aantal_elementen=count($dag);

echo "Onze array \"\$dag\" bevat $aantal_elementen elementen, zijnde het
aantal dagen  van de week <br>\n";
?>
</p>
</body>
</html>
```

Ik maak hier een nieuwe variable aan nl $aantal_elementen en de waarde daarvan wordt mij geleverd door de functie count($dag) die het aantal elementen telt in de array $dag. Die functie kan bijvoorbeeld handig zijn als je een array wilt aanmaken met gegevens uit een database, en je op voorhand niet weet hoeveel gegevens die database juist bevat.

Associatieve arrays of hashes

Nu wordt de index van een array altijd voorgesteld door middel van een getal. Hashes laten toe een element in de array te benaderen door middel van een sleutel die vrij door jouw te bepalen is. Opnieuw hebben we een korte en lange methode om die sleutels aan te maken.

De lange notatie van hashes.

We gaan dit terug demonstreren aan de hand van een voorbeeld :

voorbeeld : hashes1.php

```
<html>
<head>
<title></title>
</head>
<body>
<p>
<?php
$hoofdstad["DE"]="Berlijn";
$hoofdstad["BE"]="Brussel";
$hoofdstad["ES"]="Madrid";
$hoofdstad["DK"]="Kopenhagen";
$hoofdstad["FR"]="Parijs";
$hoofdstad["GB"]="London";

// De hoofdstad van België uitprinten

echo $hoofdstad["BE"];
?>
</p>
</body>
</html>
```

Als je nu die pagina laadt in je browser zal hij netjes Brussel tonen. Ik wil hier een hoofdstad uitprinten, dewelke wordt bepaald, nu niet meer door een getal, maar door de sleutel "BE".

De korte notatie van hashes

voorbeeld : hashes2.php

```
<html>
<head>
<title>Korte notatie van hashes</title>
</head>
<body>
<p>
<?php
$hoofdstad=array("DE"=>"Berlijn","BE" => "Brussel","ES" => "Madrid","DK" =>
"Kopenhagen","FR" => "Parijs",
"GB" => "London");

// De hoofdstad van Denemarken uitprinten

echo $hoofdstad["DK"];
?>
</p>
</body>
</html>
```

Je hebt dus terug keuze uit twee mogelijkheden, ik verkies terug de korte notatie, maar jij bent vrij deze te gebruiken die jouw het meest bevalt. Herinner je je nog wat ik zei in het begin over het niet tussen aanhalingstekens plaatsen van een variabele ? Wel het is zover. Laten we nu even voorstellen dat je wil uitprinten De hoofdstad van Spanje is Madrid. Normaal zou je dit dan zo doen :

Voorbeeld : concatenation

```
<html>
<head>
<title>Korte notatie van hashes</title>
</head>
<body>
<p>
<?php
$hoofdstad=array("DE"=>"Berlijn","BE" => "Brussel","ES" => "Madrid","DK" =>
"Kopenhagen","FR" => "Parijs",
"GB" => "London");

// De hoofdstad van Spanje uitprinten

echo "De hoofdstad van Spanje is $hoofdstad["ES"]";
?>
</p>
</body>
</html>
```

Wel dit zal niet lukken. Die aanhalingstekens rond de sleutel zijn er te veel aan. Je kunt die niet escapen. Hoe moet het dan wel ? Door gebruik te maken van de concatenation operator als volgt :

```
<html>
<head>
<title>Korte notatie van hashes</title>
</head>
<body>
```

```
<p>
<?php
$hoofdstad=array("DE"=>"Berlijn","BE" => "Brussel","ES" => "Madrid","DK" =>
"Kopenhagen","FR" => "Parijs",
"GB" => "London");

// De hoofdstad van Spanje uitprinten

echo "De hoofdstad van Spanje is " . $hoofdstad["ES"];
?>
</p>
</body>
</html>
```

Zoals je ziet, kun je op die manier ook de variabele hoofdstad["ES"] niet tussen aanhalingstekens plaatsen.

Dat hashes belangrijk zijn onder PHP zal ik later aantonen. PHP transformeert automatisch de waarden verkregen door formulieren in hashes. Dit zal duidelijk worden als we de formulieren behandelen, en dan zul je de hashes leren waarderen.

Tot zover de arrays en de variabelen.

Les 4 : Rekenen met PHP

Net zoals met iedere programmeertaal, kun je ook rekenen met PHP. Het is hier niet mijn bedoeling om jullie een cursus wiskunde te geven, we houden het hier op de eenvoudige rekenkundige bewerkingen, leerstof van het lagere onderwijs. Mochten jullie niet snappen over wat ik het heb, dan zul je wel even de wiskundige basis van het lagere onderwijs meten doornemen :-)

De rekenkundige operatoren

Heel eenvoudig, je hoeft slechts de 4 rekenkundige operatoren te gebruiken die je kent uit spreadsheets, nl : +,-,* en /

De voornaamste rekenkundige operatoren

Operator	Betekenis	Voorbeeld
+	optellen	$a=10+$b
-	aftrekken	$c=$a-1
*	vermenigvuldigen	10*3
/	delen	$a/$g
++	1 toevoegen	$a++ (is gewoon de korte schrijfwijze van $a=$a+1)
--	1 aftrekken	$a-- (equivalent aan $a=$a-1)

Verwonderd over de laatste twee elementen ? Onder PHP, net zoals onder

andere programmeertalen hebben wij hier deze bijzondere tekens ++ en --, die respectievelijk een waarde met 1 verhogen of verlagen. In klare taal :

```
$teller++
betekent hetzelfde als
$teller=$teller+1
en
$teller--
betekent hetzelfde als
$teller=$teller-1
```

Dit zijn operatoren die we veelvuldig gaan gebruiken. Onthoud dat dit 1 optelt of aftrekt bij de huidige waarde van de variabele.

Voorbeeld optellen.php :

```
<html>
<head>
<title>Optellen</title>
</head>
<body>
<p>
<?php
$a=100;
$b=35;
$resultaat=$a+$b;
echo $resultaat;
?>
</p>
</body>
</html>
```

We geven hier de waarde 100 aan de variabele a en 35 aan de vraiabele b. We tellen a en b op en kennen deze waarde toe aan de variabele $resultaat, waarna we door middel van het echo commando de waarde afprinten.

We gaan het nu iets moeilijker doen. Zoals je wellicht weet wordt in België BTW geheven op produkten. Deze BTW bedraagt 19,6%, je vindt dat misschien te hoog ? Zonder twijfel, maar het verandert niets aan onze berekening. We nemen nu als basis een produkt dat 100 Euro kost en we voegen daarbij 19,6 % BTW en we printen dan het totaal af wat het produkt kost.

Voorbeeld : btw.php

```
<html>
<head>
<title>BTW</title>
</head>
<body>
<p>
<?php
$prijs_zbtw=100;
$btwtarief=0.196;
$resultaat=$prijs_zbtw*$btwtarief;
echo $resultaat;
?>
</p>
</body>
```

```
</html
```

De funktie round()

Dit script heeft hier wel één foutje. Verander eens de $prijs_zbtw in bv 113, het resultaat wordt dan : 135,148. Nu zitten we hier met 3 cijfers na de komma, geen enkele munt ter wereld werkt met dergelijke waarden. We willen echter dat de uitkomst afgeprint wordt met 2 cijfers na de komma, wel daar hebben we de funktie round() voor. Round gebruik je als volgt round($variabele,aantal_decimalen). In os voorbeeld wordt dit dan :

Voorbeeld : round.php

```
<html>
<head>
<title>BTW</title>
</head>
<body>
<p>
<?php
$prijs_zbtw=113;
$btwtarief=0.196;
$btw=$prijs_zbtw*$btwtarief;
$resultaat=$prijs_zbtw+$btw;
echo round($resultaat,2);
?>
</p>
</body>
</html>
```

Waarna de uitkomst 135.15 wordt. Dit is dan een som die we wel met onze munt kunnen betalen.

De voorgaande voorbeelden waren vrij eenvoudig, wat zou je ervan denken eens een rekenkundige bewerking te maken waarbij we alle rekenkundige operatoren gebruiken ? We blijven in de wereld van de BTW en we gaan nu eens de prijs berekenen wat een produkt kost zonder BTW, uitgaande van de prijs BTW inclusief. Is gewoon de regel van drie toepassen. De totale waarde van het produkt is gelijk aan 100%. Dit kun je ook schrijven als 1. Het verschil tussen het bedrag met BTW en zonder BTW is het BTW percentage. Dus we moeten het BTW percentage aftrekken van de totale waarde.

Voorbeeld : btw2.php

```
<html>
<head>
<title>btw2</title>
</head>
<body>
<p>
<?php
$prijs_mbtw=119.6;
$btw_tarief=0.196;
$resultaat=$prijs_mbtw-1*$prijs_mbtw/(1+$btw_tarief);
$produkt=$prijs_mbtw-$resultaat;
echo "De prijs van het produkt zonder BTW = $produkt euro";
?>
</p>
</body>
</html>
```

Let hier wel even op het gebruik van de haakjes. Zonder deze zou PHP rekenen volgens de prioriteitsregels van de wiskundige bewerkingen, eerst vermenigvuldigen dan delen, optellen en aftrekken. Je vraagt je misschien af waarom dit zo moeilijk maken, als je de prijs met BTW vermenigvuldigt met 1.196 kom je aan hetzelfde resultaat. Inderdaad, maar dan gaat het script enkel op met een BTW percentage van 19,6 %. Op de wijze die ik hier gebruik, kun je het script ook gebruiken als de waarde van het BTW percentage anders is dan 19,6%.

Waarden afprinten met of zonder de concatenation operator punt.

Laten we nu even een test doen. We gebruiken nu terug het script btw2.php, en we willen afprinten : **De BTW op een produkt dat 119,6 euro kost is gelijk aan 19,6 euro.**. We zouden dit dan kunnen schrijven als :

Voorbeeld : btw3.php

```
<html>
<head>
<title>btw2</title>
</head>
<body>
<p>
<?php
$prijs_mbtw=119.6;
$btw_tarief=0.196;
$resultaat=$prijs_mbtw-1*$prijs_mbtw/(1+$btw_tarief);
$produkt=$prijs_mbtw-$resultaat;
echo "De BTW op een produkt dat 119,6 euro kost is gelijk aan round
($resultaat,2) euro.";
?>
</p>
</body>
</html>
```

Inderdaad, als je dat nu op deze wijze schrijft waord het resultaat in de browser :

De BTW op een produkt dat 119,6 euro kost is gelijk aan round(19.6,2) euro.

En dat willen we niet, hij moet het BTW tarief tonen en niet de funktie round. Aangezien de funktie round hier tussen de aanhalingstekens staat van het echo commando, voert PHP de funktie round niet uit, en print gewoon alles af wat er tussen de aanhalingstekens staat. Om dit op te lossen rest ons alleen de concatenation operator punt te gebruiken. Het echo commando wordt alzo :

Voorbeeld : btw4.php

```
<html>
<head>
<title>btw2</title>
</head>
<body>
<p>
<?php
$prijs_mbtw=119.6;
```

```
$btw_tarief=0.196;
$resultaat=$prijs_mbtw-1*$prijs_mbtw/(1+$btw_tarief);
$produkt=$prijs_mbtw-$resultaat;
echo "De BTW op een produkt dat 119,6 euro kost is gelijk aan " . round
($resultaat,2) . " euro.";
?>
</p>
</body>
</html>
```

En nu toont onze browser correct de zin : **De BTW op een produkt dat 119,6 euro kost is gelijk aan 19.6 euro.** Let op de spatie die ik laat na het woordje aan en voor het woordje euro. Mocht je dit niet doen, dan zou hij alles aan elkaar plakken en zou de zin er als volgt uitzien : **De BTW op een produkt dat 119,6 euro kost is gelijk aan19.6euro.**

Dit is nu een typisch geval waar je verplicht bent de concatenation operator punt te gebruiken. We hebben nu de funktie round() gezien om onze waarden af te ronden tot 2 cijfers na de komma. Er bestaat echter nog een andere funktie die dit ook kan, namelijk *sprintf()*.

De funktie sprintf()

De funktie *sprintf* wordt gebruikt als sprintf("instructie",argument). Het argument is hier duidelijk, daar plaatsen we onze variabele $resultaat in. Ik kan hier een ganse cursus vol schrijven over de funktie sprintf, maar ik zal mij hier beperken tot het omzetten van waarden in een monetair formaat, of tot 2 cijfers na de komma. De instructie die we daarvoor gebruiken is : %01.2f.

In ons voorbeeld wordt dit dan $resultaat=sprintf("%01.2f",$resultaat). Als je deze notatie gebruikt, zullen alle decimale waarden worden omgezet tot 2 cijfers na de komma. Er is hier dus een zeker verband met de funktie round (), de cijfers na de komma worden afgerond. In tegenstelling tot de funktie round(), worden bij de funktie sprintf() de mankerende getallen na de 2 decimalen aangevuld met nullen.

Voorbeeld : sprintf.php

```
<html>
<head>
<title>BTW</title>
</head>
<body>
<p>
<?php
$prijs_zbtw=113;
$btwtarief=0.196;
$btw=$prijs_zbtw*$btwtarief;
$resultaat=$prijs_zbtw+$btw;
$resultaat=sprintf("%01.2f",$resultaat);
echo "De prijs van het produkt inclusief BTW bedraagt $resultaat";
?>
</p>
</body>
</html>
```

Aan jullie nu de keus welke funktie je wenst te gebruiken. Persoonlijk vind ik round() gemakkelijker in het gebruik, het typt alvast beter :-)

Les 5 : Beheren van formulieren

In den beginne had je het formulier. Later verschenen de webwinkels, de betaalsites en andere commerciële instellingen die van jouw slechts één ding verwachten, namelijk het nummer van je kredietkaart en je e-mail adres.

Om het even, formulieren zijn zowat de uitvinding geweest van het web. Hoe zou men zonder hen de hand kunnen leggen op de gegevens van een client om die dan door te sturen naar een provider bijvoorbeeld.

De eerste versies van HTML hadden reeds formulieren, lijsten en checkbuttons geïntegreerd.

Voor de lessen die nu volgen, vertrek ik van het principe dat je over een goede basiskennis HTML beschikt. Laten we dan maar beginnen met het serieuze werk :-)

Ons eerste formulier

Voorbeeld formulier.html

```
<html>
<head>
<title>Formulier HTML</title>
</head>
<body>
<h1>Formulier met input text</h1>
<form>
Uw naam :
<input type="text" name="Naam"><br><br>
<input type="submit" value="Versturen van gegevens">
</form>
</body>
</html>
```

Als je deze pagina opgemaakt hebt en je bekijkt deze je browser, dan kun je je naam invullen en klikken op de knop versturen van gegevens, waarna ..... juist, er niets gebeurt.

Nu is het de bedoeling dat er één of andere actie wordt ondernomen als je klikt op de knop. Deze actie kan bijvoorbeeld zijn : de gegevens uitprinten op de huidige pagina of de gegevens doorsturen naar een andere pagina waar deze dan op hun beurt verwerkt worden. Aan de eerste methode heb je niet veel, de tweede is echter heel belangrijk, en die zal dan doorgaans gebruikt worden.

Laten we nu wat aktie toevoegen aan ons formulier. We passen de code aan op volgende wijze :

voorbeeld formulier2.html

```
<html>
<head>
<title>Formulier HTML</title>
</head>
<body>
<h1>Formulier met input text</h1>
<form action="verwerken.php" method="post">
Uw naam :
<input type="text" name="Naam"><br><br>
<input type="submit" value="Versturen van gegevens">
```

```
</form>
</body>
</html>
```

Het enige lijntje dat we nu veranderd hebben is de <form> regel. Als aktie hebben we gekozen voor verwerken.php en de methode om dat te doen is post. Dus als je klikt op de knop gegevens versturen, zal je naam dat je hebt ingetikt worden verstuurd naar de pagina verwerken.php.

Een andere methode in plaats van post is get. In de uitvoering is geen enkel verschil, als je nu post of get gebruikt, het resultaat zal hetzelfde zijn.

Methode : post of get ?

Wat betekent het woordje method ? Dit attribuut is verantwoordelijk voor de gebruikte methode voor het versturen van de gegevens.

Get : de gegevens die worden verstuurd, worden toegevoegd aan de url. Je kent waarschijnlijk wel die lange
url's met al die vraagtekens erin ? Get is de methode die standaard wordt gebruikt.

Post : de gegevens worden eveneens verstuurd, maar worden niet toegevoegd aan de URL.

Ik kan enkel en alleen aanbevelen om de methode post te gebruiken en get te vergeten. Voorbeelden spreken altijd voor zichzelf, wordt beweerd. Het voordeel van post tegenover get zal je dadelijk duidelijk worden aan de hand van de volgende voorbeelden.

Voorbeeld met get : passwoord1.html

```
<html>
<head>
<title>Paswoord 1</title>
</head>
<body>
<h1>Formulier met input text</h1>
<form action="verwerken.php" method="get">
Uw paswoord :
<input type="text" name="Naam"><br><br>
<input type="submit" value="Versturen van gegevens">
</form>
</body>
</html>
```

Voorbeeld met post : passwoord2.html

```
<html>
<head>
<title>Paswoord 2</title>
</head>
<body>
<h1>Formulier met input text</h1>
<form action="verwerken.php" method="post">
```

```
Uw paswoord :
<input type="text" name="Naam"><br><br>
<input type="submit" value="Versturen van gegevens">
</form>
</body>
</html>
```

De pagina verwerken.php bestaat nog niet, maak een lege pagina aan en noem die verwerken.php. Bekijk nu eens die twee pagina's in je browser, vul je paswoord in en klik dan op de knop versturen, je zult dan direkt de reden zien waarom we get niet zullen gebruiken.

In het geval van get ziet de url als volgt uit na het klikken op de knop versturen :

http://laptop.essetee.be/lesphp/verwerken.php?Naam=pass1234

Ik had hier als paswoord "pass1234" ingevuld, en je ziet het dan ook in de URL van je browser als je op de knop versturen klikt. Bij post zie je enkel de naam van de pagina waarnaar de gegevens worden verstuurd. Dus niet vergeten **STEEDS** post gebruiken.

Gegevens verwerken

Zoals gezegd in het begin van de cursus, PHP is uitermate geschikt om die gegevens te verwerken. Hier komen onze hashes eraan. De gegevens van de velden worden voor PHP gewoon variabelen. In ons geval dan met het formulier2.html wordt aan het veld "Naam" gewoon de inhoud van hetgeen je intikt gekoppeld. Je kan dus gewoon de inhoud van die variable opvragen met $_POST["Naam"].

Voorbeeld : verwerken.php broncode verwerken.php

```
<html>
<head>
<title>Gegevens verwerken</title>
</head>
<body>
<p><h1>Gegevens verwerken</h1>
</p>
<?php
echo "Goeiedag <strong>" . $_POST["Naam"] . "</strong>, leuk dat je er bent
!\n<br>";
?>
</body>
</html>
```

Als alles goed gaat moet je browser je nu goedendag zeggen als je je naam hebt ingevuld en daarna op de knop versturen hebt geklikt. Het belangrijkste is dus niet dat hij je naam kan afprinten, maar wel dat je je naam uit die variabele kan halen. Eenmaal die gegevens in je bezit, kun je die toevoegen aan een database bijvoorbeeld enz..

We gaan het nu wat moeilijker maken. Iedereen kent zijn jeugdjaren en de dreigementen van zij ouders in de aard van : Ruim onmiddellijk je kamer op, indien je het niet doet, dan geen TV vanavond :-)

Voila, dit brengt ons naar het volgende hoofdstuk :

De *if-else* structuur

De if-else structuur is één van de meest gebruikte vormen van controle in elke programmeertaal. Als de uitdrukking waar is doen we dat, in het geval die niet waar is doen we dit. Met het dreigement van onze ouders wordt dit dan :

```
Als (de kamer is opgeruimd) {
     tv kijken
}
anders {
     geen TV kijken
}
```

In PHP taal wordt dit dan :

```
if (condition) {
     A;
}
else {
     B;
}
```

Accolades gebruiken voor de blokken

Een detail springt hier duidelijk in het oog, het gebruik van de accolades. Je hebt nu steeds geleerd dat elke instructie in php wordt afgesloten met een punt-komma. Welnu, expressies, funkties, lussen en de structuren if-else worden gegroepeerd tussen accolades.

De if wordt altijd gevolgd door een conditie tussen haakjes. bijvoorbeeld if (a=10). Als A gelijk is aan 10 wordt opdracht 1 uitgevoerd, indien niet wordt opdracht 2 uitgevoerd. Tussen de accolades plaats je dan wat er moet gedaan worden. Let wel op hier, nu gebruiken we terug gewone PHP commando's en die moeten op hun beurt weer afgesloten worden met een punt-komma.

Vergeet nooit je open accolades af te sluiten met een gesloten accolade. Om dit gemakkelijk te onthouden, van zodra je een open accolade intikt, tik dan direkt de gesloten accolade ook. Werk dan verder tussen de accolades.

Ons voorbeeld is hier simpel, maar naargelang we verder gaan, zullen we dikwijls genoodzaakt zijn om eventueel nog een andere if-else structuur te openen in ons blok, of ook nog eens lussen nestelen in dat blok.

We gaan dit eens toetsen aan een voorbeeld.

Voorbeeld : Man of Vrouw, jij beslist - manvrouw.html

```
<html>
<head>
<title>Man of Vrouw - Aan jouw de keuze</title>
</head>
<body>
<p>
<h1>Man of Vrouw, aan jouw de keuze</h1><br>
</p>
<form action="verwerken2.php" method="post">
Aanspreekvorm <input type="radio" name="titel" value="m"> Mijnheer
<input type="radio" name="titel" value="w"> Mevrouw<br><br>
Uw Naam : <input type="text" name="naam"><br><br>
```

```
<input type="submit" value="Versturen">
</form>
</body>
</html>
```

En we maken nu een nieuwe pagina verwerken2.php aan met volgende inhoud :

Broncode verwerken2.php

```
<html>
<head>
<title>Gegevens verwerken</title>
</head>
<body>
<p><h1>Gegevens verwerken</h1>
</p>
<?php
if ($_POST["titel"]=="m") {
echo " Goedendag Mijnheer ";
}
else {
echo "Goedendag Mevrouw ";
}
echo "<strong>" . $_POST["naam"] . "</strong> , leuk dat je er bent !
\n<br>";
?>
</body>
</html>
```

Hoe werkt dit script nu ? In ons formulier hebben we nu radiobuttons, een tekst invoerveld en een knop versturen. De radiobuttons hebben de naam titel. Een kenmerk van een radiobutton is, dat je er maar één kunt aanvinken. In het geval dat je de radiobutton van Mijnheer aanvinkt, krijgt die dan de waarde m(an), in het andere geval krijgt die de waarde w(oman). In het tekstinvoerveld tik je een naam. PHP ziet dit dan als volgt :

```
$_POST["titel"]="m" of "w"
$_POST["naam"]="Wat jij ingetikt hebt"
$_POST["submit"]="Versturen"
```

Dan gaan we in het formulier verwerken2.php eerst na wat de variabele $_POST["titel"] bevat. Is dit een m dan printen we goedendag Mijnheer af, indien het geen m is kan het enkel en alleen een w zijn, en in dat geval printen we goedendag Mevrouw af gevolgd door de waarde $_POST["naam"], zijnde de naam die jij ingetikt hebt.

We gebruiken hier nu voor het eerst een vergelijkingsoperator, is gelijk aan. if ($_POST["titel"]=="m"). Zoals je kunt merken gebruiken we **==** en niet **=**. Aangezien de waarde m een string is, wordt die tussen haakjes geplaatst om te vergelijken. We hebben natuurlijk nog andere vergelijkings operatoren, hier zijn ze :

De vergelijkings operatoren

Operator	Betekenis
==	gelijk aan

!=	niet gelijk aan
>	groter dan
<	kleiner dan
>=	groter of gelijk aan
<=	kleiner of gelijk aan
&& of and	logische en, alle expressies moeten waar zijn
|| of or	logische of, minstens één van de expressies moet waar zijn

Ons formulier heeft nu wel nog een foutje. Stel je eens voor dat de bezoeker van je site rechtstreeks op verwerken2.php terecht komt, of dat je bezoeker vergeten of verzuimd heeft aan te duiden of hij man of vrouw is. Dat zal fouten opleveren.

De if structuur werd speciaal bedacht om die fouten te controleren. Tesamen met de funktie *isset()* zullen we controleren of alle velden in ons document werden ingevuld.

De funkie isset()

Kopieer nu verwerken2.php naar verwerken3.php en pas de pagina als volgt aan :

Broncode verwerken3.php

```
<html>
<head>
<title>Gegevens verwerken</title>
</head>
<body>
<p><h1>Gegevens verwerken</h1>
</p>
<?php
    if (isset($_POST["titel"]) && isset($_POST["naam"]) && $_POST["naam"]!
=="") {
        if ($_POST["titel"]=="m") {
                echo " Goedendag Mijnheer ";
        }
        else {
                echo "Goedendag Mevrouw ";
        }
        echo "<strong>" . $_POST["naam"] . "</strong> , leuk dat je er bent
!\n<br>";
    }
    else {
        echo "Gelieve alle velden in te vullen a.u.b.\n<br>";
    }
?>
</body>
</html>
```

Kopieer nu manvrouw.html naar formulier3.html en pas als volgt aan :

```
<html>
<head>
```

```
<title>Man of Vrouw - Aan jouw de keuze</title>
</head>
<body>
<p>
<h1>Man of Vrouw, aan jouw de keuze</h1><br>
</p>
<form action="verwerken3.php" method="post">
Aanspreekvorm <input type="radio" name="titel" value="m"> Mijnheer
<input type="radio" name="titel" value="w"> Mevrouw<br><br>
Uw Naam : <input type="text" name="naam"><br><br>
<input type="submit" value="Versturen">
</form>
</body>
</html>
```

Wat hebben we nu gedaan ? Wel, door middel van de funktie isset() controleren wij of alle velden correct zijn ingevuld. Met isset($_POST ["titel"]) controleren we of er een radiobutton is aangevinkt. Nu is het zo, dat je niet verplicht bent om een naam in te tikken. PHP zal gewoon een lege string invoegen als je dat veld niet hebt ingevuld. Om dat te verhinderen kijken we of er iets ingevuld is met $_POST["naam"]!="") wat betekent dat $_POST["naam"] niet ledig mag zijn.

Je ziet dat we hier nu verschillende operatoren gebruiken, de && operator is hier belangrijk, dit wil zeggen dat aan alle voorwaarden voldaan moet worden. Wat indien aan één voorwaarde niet voldaan wordt ? Wel, daar heb ik onderaan een else opdarcht bijgevoegd die in dit geval dan print dat je alle velden moet invullen.

Dit is hier dan ook een klassiek voorbeeld van genestelde if blokken. Zoals je ziet hou ik de blokken tesamen door tab-spaties in te voegen, zodat je weet welke IF en ELSE blok bij elkaar horen. Kijk ook naar de plaats van de accolades. Als je dit op die manier tesamen houdt zul je fouten rapper kunnen opsporen.

Kijk naar de pijltjes die ik plaatste bij de accolades.

Jammer genoeg beschikt PHP niet over een functie om je terug te laten keren naar de vorige pagina indien niet alle velden werden ingevuld. Je kunt daarvoor gebruik maken van JavaScript, maar aangezien niet alle browsers JavaScript ondersteunen, moet je dan nog eens een regel toevoegen voor deze browsers. Ik hou het simpel, je kunt natuurlijk meedelen dat de gebruiker op zijn back-button van zijn browser moet klikken, om terug te gaan naar de vorige pagina, maar ik maak gewoon gebruik van een HTML hyperlink.

Klik op de link verwerken4.php om het resultaat te zien.

Voorbeeld verwerken4.php Broncode verwerken4.php

Ik heb natuurlijk ook formulier3.html gekopiëerd naar formulier4.html en de lijn verwerken3.php vervangen door verwerken4.php

Zoals je kunt zien, hoef je het niet steeds moeilijk te maken, gewoon een HTML hyperlink zorgt dat de gebruiker terug gaat naar de pagina waarop de gegevens werden gevraagd. Als je nu Mozilla gebruikt als browser, zullen je gegevens netjes terug ingevuld staan, allez, als je bevestigend hebt geantwoord op de vraag of hij het moest onthouden wat je hebt ingevuld :-)

De $PHP_SELF variabel

We hebben nu steeds gezien dat de actie in een formulier steeds verwijst

naar een andere pagina. Wat als je nu wenst dat alles op hetzelfde formulier wordt weergegeven ? De inhoud van de $PHP_SELF variabele is altijd de naam van de huidige pagina die afgebeeld wordt in je browser.

Laat ons eens de variabele $PHP_SELF gebruiken op ons formulier3.html dat we op de volgende manier aanpassen en opslaan als formulier5.php broncode formulier5.php :

```
<html>
<head>
<title>Man of Vrouw - Aan jouw de keuze</title>
</head>
<body>
<p>
<h1>Man of Vrouw, aan jouw de keuze</h1><br>
</p>
<form action="<?php echo $PHP_SELF; ?>" method="post">
Aanspreekvorm <input type="radio" name="titel" value="m"> Mijnheer
<input type="radio" name="titel" value="w"> Mevrouw<br><br>
Uw Naam : <input type="text" name="naam"><br><br>
<input type="submit" name="versturen" value="Versturen">
</form>
<hr> <br>
<?php
    if (isset($_POST["titel"]) && isset($_POST["naam"]) && $_POST["naam"]!
=="") {
        if ($_POST["titel"]=="m") {
                echo " Goedendag Mijnheer ";
        }
        else {
                echo "Goedendag Mevrouw ";
        }
        echo "<strong>" . $_POST["naam"] . "</strong> , leuk dat je er bent
!\n<br>";
    }
    else {
        if (isset($_POST["versturen"])) {
        echo "Gelieve alle velden in te vullen a.u.b.\n<br>";
    }
    }
?>
</body>
</html>
```

Wat hebben we nu aangepast en wat is er anders ? We hebben nu onze knop versturen een naam gegeven nl :<input type="submit" **name="versturen"** value="Versturen"> Vervolgens heb ik een **<hr>
** toegevoegd op het einde van het formulier. Verder heb ik hier een nieuwe isset funktie toegevoegd in de else instructie, nl : **if (isset($_POST["versturen"])) {** en daartoe ook nog een extra accolade om deze isset af te sluiten.

Als je nu formulieren5.php oproept, gegevens invult en op de knop versturen klikt, dan zal hetgeen je ingetikt hebben verschijnen onder de horizontale lijn, in tegenstelling tot de voorbeelden die we gebruikt hebben, waar hetgeen je ingetikt had, steeds op een andere pagina werd verwerkt.

Waarom nu onze knop voorzien van een naam ? Wel, daaraan koppel ik de tweede isset funtkie in de else instructie. Door deze isset funktie wordt voorkomen dat het echo commando wordt uitgevoerd bij het openen van de pagina. Zolang je niet klikt op de knop versturen wordt het echo commando niet uitgevoerd, als je op de knop klikt en je hebt één of meerdere gegevens niet ingevuld, dan pas wordt het echo commando uitgevoerd.

Moeilijk ? Het is belangrijk dat je weet wat hier gebeurt, als we bij de paswoorden aanbelanden, zullen we hier veelvuldig van gebruik maken. Nog even samenvatten zodat je het zeker begrijpt.

Bij het openen van het formulier5.php krijg je nu gewoon de html pagina te zien zoals we al gewoon waren. We willen nu dat de gebruiker aanvinkt of ie man of vrouw is en vervolgens dat hij zijn naam intikt. We willen wel dat hij alle informatie verplicht moet invullen, zoniet, krijgt hij te zien dat hij alle velden moet invullen.

Als de gegevens zijn ingevuld kijken we door middel van de eerste isset funktie of alles is ingevuld, zo ja, dan printen we Goedendag Mijnheer/Mevrouw naam , leuk dat je er bent !. Als we nu die tweede isset funktie niet invoegen, zal dadelijk onder de horizontale lijn ook al staan : Gelieve alle velden in te vullen a.u.b. In dit voorbeeld is dit niet erg, zo weet de gebruiker dat hij alle velden moet invullen. Door middel van de tweede isset funktie gaan we na of er al op de knop is geklikt of niet. Zolang je niet klikt op de knop is de variabele $_POST ["versturen"] ledig. Op het ogenblik dat je erop klikt, krijgt hij de waarde "Versturen".

We zitten hier in een if-else lus. Dus als aan de voorwaarden van de if wordt voldaan wordt gewoon de if lus afgewerkt. Indien aan één van de voorwaarden niet wordt voldaan, dan pas wordt de else lus afgewerkt. Dus, je vinkt een optie aan, vult een naam in en klikt op de knop versturen. Aan alle voorwaarden wordt voldaan dus er wordt afgedrukt dat het leuk is om jouw te zien. Als er nu echter aan één voorwaarde niet voldaan wordt, dan is het de beurt aan de else lus om jouw er attent op te maken dat je alle velden moet invullen. Als we dit nu niet controleren via de isset funktie of je al dan niet op de knop heb geklikt, dan is er aan geen enkele voorwaarde van de if lus voldaan tijdens het openen van de pagina, en wordt bijgevolg de esle lus van toepassing, zodat hij direkt print dat je alle velden moet invullen. Door die tweede isset funktie toe te voegen, wordt tijdens het openen van de pagina noch aan de if lus, noch aan de else lus voldaan, waarop er .... inderdaad niets gebeurt.:-)

Als je nu formulier5.php runt, dan zul je zien dat de gegevens die je invulde terug verdwenen zijn als je de foutmedling krijgt dat je alle gegevens moet invullen. In ons voorbeeld moet je niet veel tikken, maar stel je voor dat je heel wat velden hebt moeten invullen, en als je dan klikt op de knop versturen, wordt je doodleuk verteld dat je iets vergeten hebt van in te vullen, en je kunt helemaal opnieuw beginnen. Ik ben het al meer dan eens tegengekomen bij het invullen van een formulier, en dat was telkens vloeken als ik moest herbeginnen.

Wel, wij willen niet dat onze bezoekers op onze site moeten vloeken, dus gaan we daar een mouw aanpassen. Je hebt nu steeds gezien dat je aan de velden in je formulier een naam en een waarde toekent. Wel we gaan nu gewoon een dynamische waarde toekennen aan onze velden. Als de pagina dan de eerste maal geladen wordt, zal de naam ledig zijn, maar van zodra we iets ingetikt hebben zal de naam de waarde aannemen van wat we ingetikt hebben. Als ze je dan terugsturen om meer gegevens in te vulllen, zullen de waarden die we reeds ingevuld hadden, netjes door PHP worden terug geplaatst.

We nemen hier terug on formulier5.php dat we kopiëren naar formulier6.php dat we dan wat gaan aanpassen :

Voorbeeld formulier6.php broncode formulier6.php

```
<html>
<head>
<title>Man of Vrouw - Aan jouw de keuze</title>
</head>
<body>
```

```
<p>
<h1>Man of Vrouw, aan jouw de keuze</h1><br>
</p>
<form action="<?php echo $PHP_SELF; ?>" method="post">
Aanspreekvorm <input type="radio" name="titel" value="m"
<?php if (isset ($_POST["titel"]) && $_POST["titel"]=="m") {echo "
checked=\"checked\"";}?>> Mijnheer

<input type="radio" name="titel" value="w"

<?php if (isset ($_POST["titel"]) && $_POST["titel"]=="w") {echo "
checked=\"checked\"";}?>> Mevrouw

<br><br>

Uw Naam : <input type="text" name="naam" value="<?php if (isset($_POST
["naam"])) { echo $_POST["naam"];} ?>" >

<br><br>


<input type="submit" name="versturen" value="Versturen">
</form>
<hr><br>
<?php
    if (isset($_POST["titel"]) && isset($_POST["naam"]) && $_POST["naam"]!
="") {
        if ($_POST["titel"]=="m") {
                echo " Goedendag Mijnheer ";
        }
        else {
                echo "Goedendag Mevrouw ";
        }
        echo "<strong>" . $_POST["naam"] . "</strong> , leuk dat je er bent
!\n<br>";
    }
    else {
        if (isset($_POST["versturen"])) {
        echo "Gelieve alle velden in te vullen a.u.b.\n<br>";
    }
    }
?>
</body>
</html>
```

Ganse boterham niet ? Zoals je nu ziet geef ik in de input lijn het veld een waarde. Wat is die waarde ? Wel eenvoudig, we gebruiken nu terug de PHP code met de funktie isset om de waarde van naam te controleren. Als je het formulier nu voor de eerste keer oproept, is die waarde gelijk aan 0 of ledig. Van zodra je nu iets intikt in het invoerveld, zal $_POST["naam"] de waarde aannemen van wat je intikt. Als je nu vergeet van een radiobutton aan te vinken, en je klikt op de knop versturen, krijg je te zien dat je alle velden moet invullen, waarop de pagina wordt herladen. Nu als de pagina wordt herladen, heeft de variabele $_POST["naam"] nog steeds de waarde van wat jij eerder hebt ingetikt, en wordt die dan ook overeenkomstig al terug ingevuld.

Hetzelfde voor de radiobuttons. De eerste maal heeft die geen waarde, maar van zodra je er ééntje hebt aangevinkt, wordt bij een reload van de pagina de gekozen radiobutton terug aangevinkt. Dat bereiken we door de waarde "checked" te specifiêren indien je er ééntje aanvinkt.

Zo zie je maar dat je gewoon PHP code dooreen de HTML syntax kunt mengen zonder problemen. Let nu wel op, de code staat wel degelijk in de input lijn, want zij maakt er deel van uit. De originele input lijn was :

Uw Naam : <input type="text" name="naam">

En de nieuwe lijn wordt dan :

Uw Naam : <input type="text" name="naam" value="<?php if (isset($_POST ["naam"])) { echo $_POST["naam"];} ?>" >

Let dus op de aanhalingstekens en de haakjes. De input lijn begint met een haakje en eindigt met een haakje.

Het begint leuk te worden, niet ? We gaan het nog leuker maken. Stel je nu eens voor dat je al die namen die ze bij jouw ingeven dienen om lid te worden van het één of het andere en dat ze dan een boekje toegestuurd krijgen. Als die persoon nu eens O 'Hara heet ? Die man kan daar wel niets aan doen, maar hij zal nooit een boekje krijgen. Probeer gewoon eens die naam in tegeven en klik op versturen, het resultaat is :

Goedendag Mijnheer **O \'Hara** , leuk dat je er bent !

Als je dit moet verwerken in een database, kun je het gewoon vergeten. Wat gebeurt hier nu ? Rembember het escapen van karakters ? Wel PHP doet dit auto voor jouw hier. Fijn van PHP, maar dit willen we niet. We willen dat O 'Hara correct in onze ledenlijst wordt opgenomen en dat zijn naam correct wordt weergegeven.

Denk ook maar eens aan een gastenboek, tikfouten zijn rap gebeurd, en van zodra je speciale tekens gaat invoeren of verkeerdelijk gaat invoeren zal de PHP code gebroken worden. Daar is wel een mouw aan te passen.

De funktie htmlspecialchars()

Om alle karakters die onder HTML een speciale betekenis hebben correct weer te geven op een HTML-pagina gebruiken we de funktie htmlspecialvars ($variabele). Deze funktie zal al deze tekens omzetten naar hun HTML equivalent.

Het karakter	wordt	en betekent
<	lt;	kleiner dan
>	gt;	groter dan
&	amp;	ampersand
"	quot;	aanhalingsteken

,
In de tweede kolom "wordt" moet je natuurlijk nog een ampersand voor de tekst plaatsen, als ik die hier plaats, dan zul je natuurlijk twee identieke kolommen te zien krijgen :-)

Voor zover dan de speciale HTML tekens, wat nu voor backslashes ? Om die correct weer te geven hebben we de speciale funktie stripslashes().

De funktie stripslashes()

Deze funktie verwijdert alle backslashes in een tekenreeks. Backslashes, dubbele backslashes die het escapen mogelijk maken van een backslashes. De syntax van de funktie is stripslashes($variabele).

Om dit nu toe te passen op onze pagina, moeten we alleen de code lichtjes aanpassen als volgt : Voorbeeld formulier7.php broncode formulier7.php

```
<html>
<head>
<title>Man of Vrouw - Aan jouw de keuze</title>
</head>
<body>
<p>
<h1>Man of Vrouw, aan jouw de keuze</h1><br>
</p>
<form action="<?php echo $PHP_SELF; ?>" method="post">
<?php
$naam=$_POST["naam"];
$titel=$_POST["titel"];
$versturen=$POST["versturen"];
?>
Aanspreekvorm <input type="radio" name="titel" value="m"<?php if (isset
($naam) && $titel=="m")
{echo " checked=\"checked\"";}?>> Mijnheer
<input type="radio" name="titel" value="w"<?php if (isset ($titel) &&
$titel=="w") {echo " checked=\"checked\"";}?
>> Mevrouw<br><br>
Uw Naam : <input type="text" name="naam" value="<?php if (isset($naam))
{$naam=htmlspecialchars($naam);
$naam=stripslashes($naam); echo $naam;} ?>" ><br><br>
<input type="submit" name="versturen" value="Versturen">
</form>
<hr><br>
<?php
    if (isset($titel) && isset($naam) && $naam !="") {
        if ($titel=="m") {
                echo " Goedendag Mijnheer ";
        }
        else {
                echo "Goedendag Mevrouw ";
        }
        echo "<strong>" . $naam . "</strong> , leuk dat je er bent !
\n<br>";
    }
    else {
        if (isset($versturen)) {
        echo "Gelieve alle velden in te vullen a.u.b.\n<br>";
    }
    }
?>
</body>
</html>
```

Ik heb hier de pagina wat leesbaarder gemaakt, zoals je kunt zien. In plaats van steeds te werken met die $_POST["variabel"] heb ik in het begin nu de waarde toegekend aan een makkelijker te manipuleren variabele.

```
<?php
$naam=$_POST["naam"];
$titel=$_POST["titel"];
$versturen=$POST["versturen"];
?>
```

Dit maakt het tikken eenvoudiger en zo zal ik in het vervolg ook mijn pagina's opmaken.

De lijn die nu opnieuw werd aangepast is :

```
Uw Naam : <input type="text" name="naam" value="<?php if (isset($naam))
```

{$naam=htmlspecialchars($naam); $naam=stripslashes($naam); echo $naam;} ?>"
>

Zoals je ziet gebruik ik eerst de htmlspecialchars op de variabele $naam en daarna de stripslashes op dezelfde variabele. Run nu opnieuw formulier7.php en vul maar eens als naam in :

s"erg/e>terr<y'n

Het resultaat zal nu zijn :

Goedendag Mijnheer **s"erg/e>terr<y'n** , leuk dat je er bent !

En dat is exact wat we intikten. Kijk nu eens naar de page source en je zult zien dat alle tekens netjes in hun HTML-equivalent werden omgezet. En nu zal O 'Hara zijn naam wel correct worden weergegeven en zal hij ook zijn boekje krijgen :-)

Nu we toch bezig zijn met tekst ziehier nog een paar funkties.

De funktie strtr()

Met deze funktie is het mogelijk om bepaalde karakters te vervangen door een ander. Stel nu, je hebt een prijslijst samengesteld en in plaats van dollars moesten het ponden zijn. Hoe we ganse teksten inlezen en veranderen zien we wat later, stel nu dat de prijs die getoond wordt $40 is maar het moest £40 zijn, wel dat doen we dan als volgt :

$prijs=strtr($prijs,"$","£");

In de variabele $prijs, veranderen we het $ teken door een £ teken.

De funktie nl2br()

nl2br staat voor newline to break, wat betekent dat als je een commentaar hoekje hebt op je site bijvoorbeeld, en iemand tikt wat commentaar in, telkens hij op de enter toets drukt om een nieuwe lijn te starten, deze automatish gaat vertaald worden naar een HTML

Even een voorbeeld : formulier8.php broncode formulier8.php

Tik nu om het even wat voor tekst in en klik dan op de knop versturen. Je zult nu zien dat de tekst letterlijk wordt weergegeven zoals je die hebt ingetikt.

Tik nu ook even in de commentaar box : Ik heb $40 betaald voor die modem. Je zult zien als je op de knop versturen klikt, je opeens £40 zult betaald hebben, heel wat duurder als je het mij vraagt :-)

Pas dus op met die funktie strtr(), hij zal overal waar hij het $ teken tegenkomt, deze vervangen door het £ teken.

De paswoorden

Paswoorden kunnen we gebruiken om bepaalde pagina's op onze website enkel toegangkelijk te maken voor de surfer met het juiste paswoord. Zoals gezegd, wordt de PHP code uitgevoerd op de server en krijgt de client de code niet te zien. Met dit in het achterhoofd, is het heel eenvoudig om paswoorden op te nemen in de webpagina, zonder daarvoor gebruik te moeten maken van een database.

Dit is dus niet de methode die aan te raden is als je zinnens bent van een grote site op te zetten, met user logins en paswoorden. Daarvoor zul je dan wel gebruik moeten maken van een database, wat hier nu buiten de les valt.

Ik wil je er tevens aan herinneren dat de apachewebserver de mogelijkheid biedt, om toegang te verbieden of te verlenen aan bepaalde users (met de nadruk op users) om bepaalde directories al dan niet te lezen. Voor meer info hierover verwijs ik jullie naar installeren van een webserver.

Paswoorden kunnen nuttig zijn als je bijvoorbeeld met meerdere mensen aan een webpagina werkt, de één kan dan een webpagina aanmaken, de andere kan die dan eventueel naar de database schrijven, een andere kan die dan verwijderen uit de database, enz ....

En wat had je gedacht van bijvoorbeeld warez of leuke prentjes ? Ik zal hier een leuk voorbeeld geven aan de hand van leuke prentjes. Een beetje lol mag ook, niet ?

We gaan nu ook onze pagina's volledig in PHP schrijven. Dan zie je ook hoe het moet. Nu heb ik jullie steeds een leeg document laten aanmaken, om vervolgens te verwerken. Wel PHP kan dat best alleen. Om dat te bereiken gaan we gebruik maken van de funktie require().

De funktie require()

Require (is nodig) is een handige funktie. Met deze funktie kun je gewoon code invoegen in een document die je aangemaakt hebt in een aparte file. Dit is enkel handig voor veel code die we gebruiken die steeds hetzelfde is. Zoals gezegd, een webpagina moet steeds het volgende bevatten :

```
<html>
<body>
</body>
</html>
```

Als je een style sheet gebruikt kun je hier die dan ook toevoegen, wat we uiteraard gaan doen :-). We gaan ook een vaste titel meegeven. Het resultaat wordt dan :

```
<html>
<head>
<title>Essetee's Hulp Pagina's</title>
<link REL="stylesheet" type="text/css" HREF="style.css">
</head>
<body>
</body>
</html>
```

Dus hebben we hier de begintags die altijd hetzelfde zullen zijn, alsook de eindtags. Wel we gaan die nu splisten en die de naam header.php en footer.php geven, met volgende inhoud :

Voorbeeld header.php

```
<?php
echo "<html>\n";
echo "<head>\n";
echo "<title>Essetee's Hulp Pagina's</title>\n";
echo "<LINK REL=\"stylesheet\" type=\"text/css\" HREF=\"style.css\">\n";
echo "</head>\n";
echo "<body>\n";
?>
```

Voorbeeld footer.php

```
<?php
```

```
echo "</body>\n";
echo "</html>\n";
?>
```

Je kunt hier mijn style.css downloaden, of er zelf ééntje aanmaken.

En nu schrijven we een pagina volledig in PHP, en voegen we in het begin met de funktie require onze header.php in, en op het einde onze footer.php. De inhoude van de pagina ziet er dan als volgt uit :

Voorbeeld require.php -- broncode require.php

```
<?php
require("header.php");
echo "<div align=\"center\"><h3>Dit wordt nu een pagina, volledig
geschreven in PHP.</h3></div><hr>\n";
require("footer.php");
?>
```

Als je nu de pagina toont in je browser, zul je nu gewoon zien : Dit wordt nu een pagina, volledig geschreven in PHP. met daaronder een horizontale lijn. Als je nu de broncode bekijkt van de pagina, zul je zien dat alle tags netjes op hun plaats staan, zoals het hoort volgens de HTML syntax.

Zoals gezegd, met die require kun je ook nog andere nuttige dingen in je pagina voegen, zoals we later gaan zien in de lessen over PHP & MySQL.

Dus de funkite require is heel eenvoudig, je brengt gegevens uit een andere file in je huidig document, en voor PHP is het dan net of dat de code zelf in de pagina wordt geschreven. Require moet je niet uitsluitend gebruiken als je een pagina volledig in PHP wenst te schrijven, zoals gebruikelijk, kun je PHP en HTML gewoon door elkaar gebruiken. Gebruiken we het vorige voorbeeld, gemengd met HTML, kan men die ook zo schrijven :

Voorbeeld require2.php -- broncode require2.php

```
<?php
require("header.php");
?>
<div align="center"><h3>Dit wordt nu een pagina, volledig geschreven in
PHP.
</h3></div><hr>

<?php
require("footer.php");
```

Als je de pagina require2.php opent in je browser, kijk dan eens naar de broncode, ik heb er commentaar bij geplaatst.

Je vraagt je nu waarschijnlijk af waarom ik voortaan de pagina's in pure PHP ga schrijven, als je ook gewoon de HTML syntax kan blijven gebruiken in comibnatie met PHP. Wel heel eenvoudig, om de source code overzichterlijker te houden. Je zult zien dat er heel wat code nodig is bij de paswoorden, en als je de pagina volledig in PHP schrijft zul je een beter overzicht houden van wat bij wat hoort.

Laten we de koe bij de horens vatten en beginnen aan ons formulier met paswoorden. We gaan vier paswoorden aanmaken, en naargelang het ingegeven paswoord, zul je een bepaalde pagina(s) kunnen bezoeken. We gaan de volgende paswoorden gebruiken :

1234	geeft toegang tot	pagina 1234.php

2345	geeft toegang tot	pagina 1234.php en 2345.php
3456	geeft toegang tot	pagina 1234.php, 2345.php en 3456.php
4567	geeft toegang tot	pagina 1234.php, 2345.php, 3456.php en 4567.php
Foutief of geen paswoord	printen we af	Geen toegang zonder geldig paswoord !

Wat springt hier dadelijk in het oog ? We hebben tot nu toe geleerd om te werken met *if-else*, m.a.w. als aan de eerste voorwaarde (if) niet wordt voldaan, dan is de tweede (else) van toepassing. Onze else opdracht hier is duidelijk, dat wordt "Geen toegang zonder geldig paswoord !". Maar wat nu met dienen if ? Wel, in PHP heeft men daar ook aan gedacht en stellen ze ons de elseif ter beschikking.

De elseif structuur

Met de elseif is het mogelijk om meerdere voorwaarden te specifiëren. In ons voobeeld wordt dat dan :

Als het paswoord gelijk is aan 1234 dan -> toegang tot pagina 1234.php of als het paswoord gelijk is aan 2345 dan -> toegang tot pagina 1234.php en 2345.php of als het paswoord gelijk is aan 3456 dan -> toegang tot pagina 1234.php, 2345 en 3456.php of als het paswoord gelijk is aan 4567 dan -> toegang tot pagina 1234.php, 2345, 3456.php en 4567.php anders -> geen toegang zonder geldig paswoord

Zoals je kunt zien is het gebruik van de *elseif* oneindig. Al de elseif opdrachten zullen doorlopen worden totdat er iets gevonden wordt met wat er werd gevraagd, en indien niets gevonden wordt dat correspondeert, wordt dan de else van toepassing. In PHP code wordt ons voorbeeld dan :

```
If ($paswoord == "1234") {
        toegang tot ....
        }
        elseif ($paswoord == "2345") {
        toegang tot ...
        }
        elseif $paswoord == "3456") {
        toegang tot ...
        }
        elseif $paswoord == "4567") {
        toegang tot ...
        }
else {
       echo "Geen toegang zonder geldig paswoord !";
}
```

Nu de praktijk. Ik heb hier vier foto's van een leuke madam. De foto's onthullen meer en meer van de madam naargelang het juiste paswoord. Voor kinderen van 1 to 5 jaar reserveren we het paswoord 1234, van 5 to 10 jaar het 2345 paswoord, van 10 to 15 jaar het 3456 paswoord en voor de rest het paswoord 4567. In eerste instantie hou ik het simpel. Ik vraag niet naar de leeftijd van de persoon, ik hou enkel rekening met het ingevoerde paswoord denkende dat jullie wel het juiste paswoord aan de juiste persoon hebben gegeven. :-)

Als je dit onder de knie hebt, dan gaan we dan de leeftijd en het paswoord

vragen. Zoals je zult zien komen er dan een massa mogelijkheden bij. vb goed paswoord, verkeerde leeftijd of omgekeerd enz ...

Laten we eerst ons formulier opmaken dat we logmetpas.php gaan noemen -- broncode logmetpas.php.

```
<?php
require("header.php");
?>
<?php
$paswoord=$_POST["paswoord"];
echo "<h1><font color=\"#2208cc\"> Gelieve in te loggen met Uw
paswoord</font></h1>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "<p><input type=\"text\" name=\"paswoord\"></p>\n";
echo "<input type=\"submit\" name=\"versturen\" value=\"Paswoord
versturen\">\n";
echo "</form><br>\n";
if (isset($paswoord)) {
        if ($paswoord=="1234") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        }
        elseif ($paswoord=="2345") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        }
        elseif ($paswoord=="3456") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
```

```
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";
        }
        elseif ($paswoord=="4567") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"4567.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar de leukste madam\">\n";
        echo "</form>\n";
        }
else {
        echo "<h3><font color=\"#db2408\">Leuk geprobeerd, maar ....
</h3>\n";
        echo "<h3><font color=\"#db2408\">Geen toegang zonder geldig
paswoord !</h3>\n";
        }
}
?>
<?php
require("footer.php");
?>
```

Voila, een ganse boterham ineens he. Feitelijk is het script erg simpel. We kijken welk paswoord is ingevoerd en aan de hand van het ingevoerde paswoord laat je de desbetreffende pagina's zien waar hij recht op geeft. In het eerste geval kan hij enkel 1 pagina zien, in het tweede geval kan hij pagina 1 en 2 zien, en zo verder. Het enige wat we eigenlijk doen in de else-if structuur, is gewoon de inhoud overnemen van de vorige if-else en dan telkens een nieuwe optie toevoegen.

Nu is er eigenlijk nog iets mis met ons script. Als het paswoord correct is ingevoerd, nemen we nu het geval 1234, dan wordt de pagina 1234.php getoond. Als je nu het script runt met dit paswoord, zie je in de naviagatiebalk van je browser hetb adres : **http://essetee.mine.nu:7080/lesphp/1234.php**. Als iemand nu dat adres kent, en intikt in zijn navigatiebalk, zal hij ook deze pagina kunnen zien. Aangezien we hier gevoelige informatie willen verschaffen aan iemand met het juiste paswoord, zullen we hieraan iets moeten doen.

Een belangrijke lijn in ons logmetpas.php script is deze :

echo "<input type=\"hidden\" name=\"paswoord\" value=\"$paswoord\">\n";

Deze lijn is van het type hidden, dwz je krijgt dit niet te zien op jouw html-pagina. We zorgen er met dit lijntje voor dat je ingegeven paswoord mede wordt verzonden naar de pagina 1234.php (voor het paswoord 1234 natuurlijk). Nu moeten we enkel controleren of er een paswoord is ingevoerd als we de pagina 1234.php opvragen. En daarvoor gebruiken we natuurlijk terug de funktie isset(). Ik ga hier enkel 1234.php aanpassen, voor de andere paginas laat ik het aan jullie over. Kopieer nu logmetpas.php naar logmetpas-1.php en 1234.php naar 1234-1.php.

Voorbeeld logmetpas-1.php broncode logmetpas-1.php

```
 <?php
require("header.php");
?>
<?php
$paswoord=$_POST["paswoord"];
echo "<h1><font color=\"#2208cc\"> Gelieve in te loggen met Uw
paswoord</font></h1>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "<p><input type=\"text\" name=\"paswoord\"></p>\n";
echo "<input type=\"submit\" name=\"versturen\" value=\"Paswoord
versturen\">\n";
echo "</form><br>\n";
if (isset($paswoord)) {
        if ($paswoord=="1234") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234-1.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        }
        elseif ($paswoord=="2345") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        }
        elseif ($paswoord=="3456") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
```

```
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";
        }
        elseif ($paswoord=="4567") {
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"4567.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar de leukste madam\">\n";
        echo "</form>\n";
        }
else {
        echo "<h3><font color=\"#db2408\">Leuk geprobeerd, maar ....
</h3>\n";
        echo "<h3><font color=\"#db2408\">Geen toegang zonder geldig
paswoord !</h3>\n";
        }
}
?>
<?php
require("footer.php");
?>
```

Ik heb hier dus enkel het lijnjte **`echo "<form action=\"1234-1.php\" method=\"post\"\n>";`** vervangen, aangezien we hier de pagina 1234-1.php gaan tonen nu i.p.v. 1234.php. Ik doe dit hier enkel omdat jullie het script zouden kunnen volgen. Je kunt gewoon logmetpas.php blijven gebruiken, doch je zult dan de pagina's die moeten worden getoond beveiligen op dezelfde manier als ik hier doe.

Het beveiligen van de pagina 1234-1.php broncode 1234-1.php

```
<?php
require("header.php");
```

```
if (isset($_POST["paswoord"]) && $_POST["paswoord"] !="" ) {
echo "<div align=\"center\"><h3>Dit is de pagina voor het paswoord 1234,
2345, 3456 en 4567</h3></div><hr>\n";
echo "<div align=\"center\"><strong>En dat is een leuke madam :-)
<br><br>\n</strong></div>";
echo "<div align=\"center\"><img
src=\"http://www.essetee.be/images/madam1234.jpg\" width=\"90\"
height=\"120\"
></div>\n";
}
else {
echo "<p><h2>Leuk geprobeerd, maar ......</h2></p>";
echo "<p><h2>Geen toegang zonder geldig paswoord</h2></p>";
<hr>
}
require("footer.php");
?>
```

Hier kijken we dus gewoon met de funktie isset() of er een paswoord is ingegeven. Als het paswoord ledig is, dan krijg je terug de melding dat het leuk geprobeerd is. Je hoeft hier niet ts specifieren of het paswoord wel correct is voor deze pagina. Dat doen we al met het script logmetpas.php. Als je daar een verkeerd paswoord hebt ingegeven, kun je nooit op deze pagina geraken. Als je nu in je navigatiebalk het adres **http://essetee.mine.nu:7080/lesphp/1234-1.php** intikt, is in dit geval het paswoord een lege string, en zul je de pagina niet kunnen zien.

Wat jullie nu egenlijk maar hoeven te doen, is in de pagina's 1234 tot en met 4567.php, gewoon die isset commando's toevoegen, en al die pagina's zijn beschermd met een paswoord. Ik doe dit met logmetpas-1.php en 1234-1.php, aangezien ik jullie wens te wijzen op de kleine veranderingen naarmate we het script aanpassen.

Nu is er nog iets mis met ons script. Zoals je ziet komt er al heel wat code bij te kijken voor 4 users met een paswoord. Stel je nu eens voor dat je 100 users hebt. Dan ben je een ganse week bezig met de code te schrijven en te verbeteren, want hoe meer code, hoe meer kans op een parse-error. Dat kun je natuurlijk oplossen door al je users in een database te steken of op te nemen in een tekstfile. Dat gaan we later dan eens doen als we aanbeland zijn bij het lezen van en schrijven naar tekstfiles onder php en als ik tijd heb, MySQL en PHP.

Deze vorm van paswoorden is goed bijvoorbeeld voor een paar users toegang te verlenen om de site wat aan te passen, of om een database bij te houden. In principe zul je geen 100 dergelijke users zulke taken laten verrichten op jouw site :-)

Begrijp nu al wat de elsif structuur ? Dan is het tijd om zijn broertje te leren kennen, de switch() funktie.

De funktie switch()

Feitelijk is de switch() funktie gelijk aan de elseif. Alleen de schrijfwijze is wat anders, maar persoonlijk gebruik ik liever switch() dan elseif. We nemen nu ons zelfde voorbeeld als voorheen. Met het juiste paswoord kun je een bepaalde pagina zien. Onder switch() wordt dit dan :

```
switch($paswoord) {

in het geval van 1234 :
                mag je kijken naar 1234.php

break;
in het geval van 2345 :
```

```
                mage je kijken naar 1234 en 2345.php

break;
 enz ....

default :
                geen toegang zonder geldig paswoord

}
```
In PHP taal wordt dit dan :
```
switch($paswoord)  {

case "1234" :
        wat je mag doen
break;
case "2345" :
        wat je mag doen
break;
case "3456" :
        wat je mag doen
break;
case "4567" :
        wat je mag doen
break;
default :
        wat er moet gebeuren als aan geen enkele case wordt voldaan
}
```
Je ziet hier ook telkens het woordje break staan na iedere case. Deze break laat je toe om de switch lus te verlaten, van zodra aan de case wordt voldaan. Dit verhindert dat het switch blok verder wordt gelezen. Mocht je bv een switch blok hebben met 1000 cases, is het goed dat ie het switchblok verlaat zodra hij een case heeft gevonden die correspondeert met de input van de user. Dit levert tijdwinst op. Maar genoeg theorie, laten we een voorbeeld opmaken met switch.

Voorbeeld logmetpas2.php broncode logmetpas2.php

```
<?php
require("header.php");
?>
<?php
$paswoord=$_POST["paswoord"];
echo "<h1><font color=\"#2208cc\"> Gelieve in te loggen met Uw
paswoord</font></h1>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "<p><input type=\"text\" name=\"paswoord\"></p>\n";
echo "<input type=\"submit\" name=\"versturen\" value=\"Paswoord
versturen\">\n";
echo "</form><br>\n";
if (isset($paswoord)) {
        switch($paswoord) {

        case "1234":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
```

```
        break;

        case "2345":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";

        break;

        case "3456":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";

        break;

        case "4567":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
```

```
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"4567.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar de leukste madam\">\n";
        echo "</form>\n";

        break;

        default:
        echo "<h3><font color=\"#db2408\">Leuk geprobeerd, maar ....
</h3>\n";
        echo "<h3><font color=\"#db2408\">Geen toegang zonder geldig
paswoord !</h3>\n";
        }
}
?>
<?php
require("footer.php");
?>
```

Zoals je kunt zien zijn hier de if en elseif vervangen door case. Voor de rest is er weinig veranderd. Alleen voeren we hier nu na de isset funktie de switch funktie in (switch($paswoord)). Verder zie je dat de woorden case gevolgd worden door een : (dubbel punt) i.p.v. de accolades bij een if blok. Het switch blok begint met een accolade en wordt afgesloten met een accolade. Je ziet hier dan ook maar 4 accolades, daarom dat ik deze vorm prefereer boven de elseif structuur. Een accolade is rap vergeten :-)

Als je nu de pagina's hebt beveiligd die mogen gezien worden, zal de switch structuur hier niets aan veranderen. Als je de url intikt van de pagina, zal deze ook niet getoond worden zonder geldig paswoord.

Een ander praktisch voorbeeld van de switch funktie. Zou het niet fijn zijn als de surfers de background kleur van de pagina kunnen veranderen ? Ziehier dan terug een staaltje PHP gemengd met HTML.

Voorbeeld achtergrondkleur veranderen : back.php -- broncode back.php

```
<html>
<head>
<title> De achtergrondkleur van de pagina veranderen </title>
</head>
<body bgcolor="<?php
if(isset($color)) {
switch($color) {
case "1":
        echo "#ffcc33";
break;
case "2":
        echo "ffff99";
break;
default:
        echo "silver";
}
}
else {
echo "white";
}
```

```
?>">
<h1> De achtergrondkleur van een pagina kiezen ! </h1>
<form action="<? echo $PHP_SELF; ?>" method="post">
<input type="radio" name="color" value="1"> Oranje<br>
<input type="radio" name="color" value="2"> Geel<br>
<input type="radio" name="color" value="x"> Verassing<br><br>
<input type="submit" value="Kleur toepassen">
</form>
</body>
</html>
```

Zoals je kunt zien gebruik ik hier voor de verassingskleur de waarde X. Als je deze waarde kiest, heb je niet voor 1 of 2 gekozen, en wordt de default dan toegepast. Als je de pagina voor de eerste keer opvraagt, zal de achtergrond in het wit zijn, zoals bepaald door de isset funktie. Bij de het eerste bezoek aan de pagina is de variabel $color leeg, en in dit geval zal de achtergrond wit zijn.

In normale HTML-taal is de body tag zo : <body bgcolor="waarde">. Zoals je kunt merken, vervang ik die waarde door mijn PHP script. Je php script moet dan ook eindigen met de ">. Je kunt het script natuurlijk op één lijn schrijven, maar dat leest niet makkelijk. Je kunt het script dan ook zo schrijven zoals ik het doe. Let wel, gebruik hier geen spaties en lege regels om de boel leesbaarder te houden, dat kan soms tot rare resultaten leiden als je het toch doet.

Zo, to zover de paswoorden. Ik zal later nog eens op die paswoorden terug komen als we files gaan inlezen en wegschrijven.

Nog een opmerking. We hebben nu steeds onze paswoorden in klare taal ingetikt. Iedereen kan dus ook zien welk paswoord je intikt. Wil je nu dat er enkel sterretjes verschijnen in plaats van je paswoord in klare taal, wel vervang gewoon echo " <input type=\"text\" name=\"paswoord\"> \n"; door echo " <input type=\"password\" name=\"paswoord\"> \n"; en de klus is geklaard. Dit is natuurlijk basic HTML-stuff, maar in de lessen kan het handiger zijn om te zien wat je intikt.

Werken met lussen onder PHP

Hopelijk hebben jullie de vorige lessen goed onder de knie, want nu begint het zware werk maar ook het plezantste. Nu dat we eenmaal bij de lussen zijn aanbeland zullen we wat spelletjes spelen en wat denk je van eens de computer lottonummers voor jou te laten trekken. Met de lussen wordt dit allemaal mogelijk.

Wat zijn lussen ?

Stel je eens de volgende situatie voor. Je zit in een kamer en de enige mogelijkheid om die kamer te verlaten, is via een glazen deur, waarachter een bewaker zit. Deze bewaker heeft niets anders te doen dan daar te zitten, en hoge studies moet je daarvoor niet hebben gedaan, met andere woorden, het is een analfabeet. Aan de muren in de kamer waar je zit, hangen bordjes met cijfers erop van 1 tot 100. De analfabeet heeft ook dergelijke bordjes, en trekt er willekeurig eentje uit. Nu moet jij een bordje van de muur nemen, naar de glazen deur gaan, en het bordje tonen aan de bewaker, die het zorgvuldig met het zijne vergelijkt. Klopt het niet, dan moet jij je bordje terug hangen, een ander nemen en terug gaan tonen aan de bewaker. Van zodra jouw bordje met het zijne klopt, opent die de deur voor jou.

Dat brengt ons bij onze eerste lus, de do ... while.

De do-while lus

```
Doe {
        toon het bordje aan de bewaker;
        }
zolang (jouw bordje niet gelijk is aan dit van de bewaker - hang je bordje
terug en toon een ander);
        De bordjes zijn gelijk en de deur gaat open;
```

Stel nu dat de bewaker het bordje heeft met het nummer 34 op ($zijnnummer). Onze lus do...while wordt dan onder PHP :

```
<?php
do {
        toon een bordje aan de bewaker;
        }
while ( jouw bordje != $zijnnummer );
        jouw bordje is gelijk, de deur gaat open;
?>
```

Nu is ons nummer dat we bij hebben 100 ($onsnummer) en schrijven we het nu in correcte PHP taal :

```
<?php
do {
        $nummer = $onsnummer;
        }
while ($nummer != $zijnnummer);
        hij opent de deur;
?>
```

Zoals je kunt zien, geven we hier de variabele de waarde van ons nummer, zijnde 100. De while vergelijkt dan de waarde van ons nummer met dit van de bewaker, en enkel en alleen indien ons nummer overeenkomt komt met het zijne, wordt de opdracht "hij opent de deur" uitgevoerd. Dus aan ons om opnieuw de variabele $nummer van een ander getal te voorzien.

Stel je nu eens voor, dat er daar een plezante jongen rondloopt, die tussen de bordjes van de bewaker er ééntje heeft gestoken met de letters AD op. Dan bestaat er maar één mogelijkheid om uit die kamer te geraken, namelijk door de glazen deur stuk te trappen. (onder unix ctrl+c en onder windows ctrl+alt+del). Dit noemen ze dan een oneindige lus, die enkel kan verlaten worden door actie van de gebruiker van het programma.

Nu, dat lijkt allemaal plezant, maar het is een veel voorkomende fout van beginnende programmeurs. Als je appels met peren vergelijkt kun je dit tegenkomen. Daarom dat je goed moet nadenken bij het opstellen van een do-while lus. In de praktijk is dat een lus die weinig wordt gebruikt.

Nu gaan we een teller invoeren. Onder de lussen is dat de normaalste zaak van de wereld. Je zult veel met tellers werken. Ze zijn niet weg te denken uit de lussen. Om op ons voorbeeld terug te keren, we voeren nu een teller in, in de do instructie, kennen de waarde van de teller toe aan $onsnummer en vergelijken dan $onsnummer met $zijnnummer. Als het nummer niet klopt, dan verhogen we de teller met 1 en beginnen we opnieuw. Als het nummer dan klopt, tonen we het bericht "De nummers stemmen overeen !".

Voorbeeld do-wile lus : dowhile.php -- broncode dowhile.php

```
 <?php
require("header.php");
# het getal van de bewaker = 34

$zijnnummer=34;
```

```
do {
        $onsnummer=$teller;
        $teller++;
        }
while ($onsnummer != $zijnnummer);
        echo "De nummers stemmen overeen<br>\n";
require("footer.php");
?>
```

We gaan nu nog een stapje verder. We laten nu ook afprinten hoeveel maal wij weg en weer hebben gelopen tot we het juiste nummer hadden. Wij gaan dan printen "Na zoveel pogingen stemden de nummers overeen !"

Voorbeeld dowhile2.php -- broncode dowhile2.php

```
<?php
require("header.php");
# het getal van de bewaker = 34

$zijnnummer=34;
do {
        $onsnummer=$teller;
        $teller++;
        }
while ($onsnummer != $zijnnummer);
        echo "Na $teller pogingen stemden de nummers overeen<br>\n";
require("footer.php");
?>
```

Nu, wat zie je ? Wel dat we 35 pogingen nodig hadden om het juiste nummer te vinden. Wel dat klopt niet, aangezien de bewaker zijn nummer 34 is, en wij dus ook maar 34 pogingen nodig moeten hebben om het nummer te raden. Zien jullie waar de fout ligt ? Indien ja, well done indien nee, hier de uitleg. Nadat we ons nummer gelijk hebben gesteld met de teller, verhogen we nadien de teller met 1. Dus, als de teller 34 is klopt ons nummer met zijn nummer, maar de teller wordt alsnog met 1 verhoogd voordat hij bij de while instructie komt.

Je kunt dit natuurlijk oplossen door $teller aan te passen als "Na $teller-1 pogingen ....." en dan zal alles goed staan. Maar dit is niet de juiste weg. De juiste weg is om een tweede teller toe te voegen, die de waarde van de eerste teller krijgt voordat die verhoogd wordt. Nu werken we nog met vaste getallen, maar denk eens na als je getallen of variabelen gaat gebruiken die uit een bestand of database komen ? Dan weet je op voorhand de waarden niet.

Dit is een gouden tip : schrijf je programma's, zodat die kunnen werken met eender welke vorm van gegevens. Dus om dat hier correct op te lossen voegen we een tweede teller bij die de pogingen onthoud. De veranderingen met het vorige script worden vet en in het rood afgedrukt.

Voorbeeld dowhile3.php -- broncode dowhile3.php

```
<?php
require("header.php");
# het getal van de bewaker = 34

$zijnnummer=34;
do {
        $onsnummer=$teller;
        $poging=$teller;
        $teller++;
        }
while ($onsnummer != $zijnnummer);
        echo "Na $poging pogingen stemden de nummers overeen<br>\n";
require("footer.php");
```

```
?>
```
Als je het script nu runt, moet hij nu tonen dat we na 34 pogingen de nummers correct hadden. Dus ik voer hier een tweede teller in die ik poging noem. Poging wordt gelijkgesteld met de waarde van de teller, maar als we nadien de teller met 1 verhogen, zal de waarde van poging niet worden verhoogd, en zal poging maar de huidige waarde van de teller aannemen als we de do instructie terug uitvoeren.

Om onze pogingen af te printen, gebruiken we dan ook niet meer de waarde van de $teller maar de waarde van $poging wat dan het correcte resultaat oplevert.

Zoals ik jullie zei, met die tellers gaan we nog veel plezier beleven. Kijk ook goed naar de plaats waar je een teller plaatst in het script. Stel je eens voor dat je $teller++; als eerste instructie plaatst in de do routine. Als de waarde die we zoeken O (nul) is, heb je dan meteen prijs. Je eerste eindeloze lus is dan gecreëerd. Zoals ik reeds zei in het begin van de cursus, ben je onder PHP niet verplicht van je variabelen op voorhand te definiëren. Hier gebruiken we voor de eerste maal $teller, en de waarde van $teller wordt dan ook door PHP op 0 (nul) geplaatst. Als je nu direkt $teller++; doet in plaats van na de instructie die $onsnummer de waarde van $teller geeft, kan $onsnummer dus nooit 0 (nul) zijn. Bijgevolg zal de enige manier zijn om het script te onderbreken de bekende toetsaanslagen zijn.

Zoals gezegd, je moet de variabelen op voorhand niet defiëneren. Maar hou er rekening mee dat een computer begint te tellen vanaf 0 (nul) en niet vanaf één zoals wij. In feite werd de do lus wel degelijk 35 maal doorlopen, maar in ons voorbeeld speelde dat feitelijk geen rol. Het correcte script om te voorkomen dat 0 (nul) wordt gebruikt is dan zo :

```
<?php
require("header.php");
# het getal van de bewaker = 34

$teller=1;
$zijnnummer=34;
do {
        $onsnummer=$teller;
        $poging=$teller;
        $teller++;
        }
while ($onsnummer != $zijnnummer);
        echo "Na $poging pogingen stemden de nummers overeen<br>\n";
require("footer.php");
?>
```
We vertellen op deze manier aan de computer dat ie moet beginnen te tellen vanaf 1, zoals wij gewoon zijn van doen.

Hopelijk begrijpen jullie dit goed, want we gaan nu ons script verder uitbreiden. Nu heeft onze bewaker het nummer 34, omdat wij het hem hebben gegeven. Fijn is echter om de bewaker zijn eigen nummer te laten trekken, en dan te kijken welk nummer hij had. Daarvoor beschikt PHP over de funktie srand.

De funktie rand()

Met de funktie rand (random) is het mogelijk om random getallen te laten genereren door de computer. Rand neemt als argument een minimum waarde en een maximumwaarde. vb rand(minimum, maximum). Wil je nu een willekeurig getal laten trekken dat tussen 1 en 100 ligt, roep je rand aan als volgt : **rand(1, 100).**

In ons voorbeeld laten we de bewaker een nummer trekken tussen 1 en 100 en kennen we dit toe aan de variabele $zijnnummer. Om dit te bereiken gaan we als volgt te werk :

Voorbeeld dowhile4.php -- broncode dowhile4.php

```
<?php
require("header.php");
# het getal van de bewaker laten trekken door de computer

$zijnnummer=rand(1, 100);
$teller=1;

do {
        $onsnummer=$teller;
        $poging=$teller;
        $teller++;
        }
while ($onsnummer != $zijnnummer);
        echo "Na $poging pogingen stemden de nummers overeen<br>\n";
require("footer.php");
?>
```

Als je nu klikt op de link dowhile4.php zul je te zien krijgen welk getal de computer heeft getrokken, namelijk het aantal pogingen die we nodig hadden om het te vinden. Als je op de reload button van je browser klikt, zal een nieuw nummer worden getrokken.

Hopelijk heb je nu al iets bijgeleerd hoe lussen werken. Onze pagina doet nu nog niet veel, maar dat gaat rap veranderen. We gaan nu namelijk de volgende lus aanpakken, de for-lus. We laten de do-while voor wat ze is, want in hoofdzaak gaan we de for lus gebruiken.

Voor ik verder ga met de for lus, nog even dit, er bestaat ook een vereenvoudigde do-while lus, namelijk de while lus. Aangezien ik die lus zelden gebruik, en jullie zullen die ook waarschijnlijk niet veel gebruiken, wil ik daar niet verder op in gaan. Ik geef wel een voorbeeldje. Stel je wilt bv 10 maal je naam afprinten op de HTML-pagina, met de while lus gaat dit als volgt :

De while lus

Voorbeeld while.php -- broncode while.php

```
<?php
require("header.php");

# het aantal op 10 plaatsen, zijnde het aantal
# malen dat we onze naam willen afprinten

$aantal=10;
$teller=1;

# Zolang de teller kleiner is dan het aantal
# Voeren we de code uit tussen de accolades.
# Na het printen van de naam, wordt de teller
# met 1 verhoogd

while($teller <= $aantal) {
echo "Serge Terryn<br>\n";
$teller++;
}

require("footer.php");
```

```
?>
```
Hier terug hetzelfde met onze teller, indien je de variabel $teller niet op 1 plaatst, zal hij onze naam 11 maal afdrukken i.p.v. de 10 maal die we wensten.

Ik voel het aan mijn ellebogen dat jullie wel eens onze bewaker aan het werk willen zien in een while lus. Hier gaan we dan :

Voorbeeld while2.php -- Broncode while2.php

```
<?php
require("header.php");
# het getal van de bewaker laten trekken door de computer

$zijnnummer=rand(1, 100);
$teller=1;
while($teller != $zijnnummer) {
        $poging=$teller;
        $teller++;
}
echo "Na $poging pogingen werd het nummer gevonden<br>\n";
require("footer.php");
?>
```
Zo het voorbeeld spreekt voor zichzelf en laten we nu beginnen aan de lussen die we het meest zullen gebruiken.

De for lus.

De for-lus wordt aangemaakt met drie condities, gescheiden door een punt-komma. Gescheiden door een punt-komma;gescheiden door een punt-komma; gescheiden door een punt-komma; gescheiden door een punt-komma; gescheiden door een punt-komma; gescheiden door een punt-komma; gescheiden door een punt-komma; gescheiden door een punt-komma.

Ik denk dat ik duidelijk ben geweest, de condities worden gescheiden door een punt-komma. Als je een for-lus aanmaakt, zal ie steeds zoeken naar die twee punt-komma's. Ook al heb je één van die condities niet nodig, de punt-komma moet er staan !

De for-lus wordt als volgt geschreven :

```
for (variabel_nummer; conditie; tellen) {
        code tot zolang aan de conditie wordt voldaan
}
```
Uit ons voorbeeld van de bewaker wordt dit :

```
for ($teller=1;$teller <= $zijnnumer;$teller++) {
        $poging=$teller;
}
```
Nu in de praktijk. Voorbeeld forlus.php -- broncode forlus.php

```
 <?php
require("header.php");
# het getal van de bewaker laten trekken door de computer

$zijnnummer=rand(1, 100);

for($teller=1;$teller <= $zijnnummer;$teller++) {
        $poging=$teller;
}
echo "Na $poging pogingen werd het nummer gevonden<br>\n";
require("footer.php");
?>
```
Ik kom terug op die punt-komma's. Zoals gezegd moet je onder PHP je

variabelen niet op voorhand definïeren. Als ik de $teller niet op 1 plaats, zal PHP voor mij de waarde 0 (nul) toe kennen aan $teller. Dus ik kan de for-lus schrijven zonder de $teller=1. Ik doe dit dan als volgt :

Voorbeeld forlus2.php -- broncode forlus2.php

```
<?php
require("header.php");
# het getal van de bewaker laten trekken door de computer

$zijnnummer=rand(1, 100);

for($teller <= $zijnnummer;$teller++) {
        $poging=$teller;
}
echo "Na $poging pogingen werd het nummer gevonden<br>\n";
require("footer.php");
?>
```

Als je dit nu bekijkt heb je direkt prijs, je krijgt namelijk te zien **Parse error: parse error, expecting `';'' in / home/serge/homepage/lesphp/forlus2.php on line 7** , daar heb je hem, de punt-komma.

Hoe moet het wel : voorbeeld forlus3.php -- broncode forlus3.php

```
<?php
require("header.php");
# het getal van de bewaker laten trekken door de computer

$zijnnummer=rand(1, 100);

for(;$teller <= $zijnnummer;$teller++) {
        $poging=$teller;
}
echo "Na $poging pogingen werd het nummer gevonden<br>\n";
require("footer.php");
?>
```

Als je dit nu bekijkt zul je geen errors meer krijgen. Ik laat de $teller=1 weg, maar ik plaats wel de punt-komma waarna ik de conditie invul, terug een punt-komma en dan de teller met 1 verhoog. Het kan gebeuren dat je een conditie niet nodig hebt, vergeet dus wel de punt-komma niet.

En nu dan de laatste en geniaalste for-lus die zijn introductie deed sedert PHP4, namelijk de foreach-lus.

De foreach lus.

Foreach loopt doorheen een array en geeft de waarden terug die erin zijn geregistreerd. Foreach kun je gebruiken onder 2 vormen :

1. De verkorte vorm zonder sleutel

```
voor elke (element in de array -met- waarde) {
        toon de waarde van het array(element);
}
```

In Nederlandse taal, doorloop de array, kijk naar de waarde en print die vervolgens af. Onder PHP schrijven we dit als :

```
foreach ($array as $value) {
        echo "$value
\n;
}
```

Herinner je je nog onze arrays uit les 3. We gaan nu terug een voorbeeld

aanmaken aan de hand van de dagen van de week.

Voorbeeld foreach.php -- broncode foreach.php

```
 <?php
require("header.php");

# Onze array definiïeren

$dag[0]="Zondag";
$dag[1]="Maandag";
$dag[2]="Disndag";
$dag[3]="Woensdag";
$dag[4]="Donderdag";
$dag[5]="Vrijdag";
$dag[6]="Zaterdag";

foreach ($dag as $value) {
        echo "$value<br>\n";
}
require("footer.php");
?>
```

Als je dit nu runt, zouden alle dagen van de week netjes moeten worden afgeprint. Dus wat doet foreach eigenlijk ? Hij doorloopt onze array $dag en geeft alle waarden terug die hij erin terug vindt.

2. De lange vorm met sleutel.

Hier zijn we dan terug met onze hashes. Deze vorm van foreach doorloopt een hash en toont daarbij elk element en zijn sleutel. De syntax is als volgt :

```
foreach ($array as $key => $value) {
        echo "$key=$value
\n";
}
```

Zie het als volgt, loop doorheen de $array, kijk naar de sleutel en print de waarde af. We gaan nu terug het voorbeeld maken met onze hoodsteden uit les 3.

Voorbeeld foreach2.php -- broncode foreach2.php

```
<?php
require("header.php");

# Onze array definiïeren

$hoofdstad["DE"]="Berlijn";
$hoofdstad["BE"]="Brussel";
$hoofdstad["ES"]="Madrid";
$hoofdstad["DK"]="Kopenhagen";
$hoofdstad["FR"]="Parijs";
$hoofdstad["GB"]="London";

foreach ($hoodstad as $key => $value) {
        echo "$key=$value<br>\n";
}

require("footer.php");
?>
```

Als je dit nu runt, worden alle hoodsteden met hun sleutel en waarde afgedrukt. Ik hoop dat jullie het belang inzien van de foreach lus. PHP is een webtaal, dus we gaan formulieren aanmaken, gegevens verwerken, gegevens doorsturen enz ... Aangezien we op voorhand niet weten wat een gebruiker

zal invullen, totdat hij het ons opstuurt, halen we gewoon de informatie met de foreach lus terug uit een bestand.

Als je foreach gebruikt zoals in ons tweede voorbeeld, kan dit een hulp zijn tijdens het programmeren. Run gewoon de foreach lus en je krijgt alle waarden van de array terug, mocht je soms vergeten zijn wat je er nu in had geplaatst, of indien de uitslag iets anders is dan je had verwacht.

Allez, we gaan terug het spelletje spelen met onze bewaker. Voor het gebruik van de foreach lus moeten we eerst onze array definïeren. Dit is hier gemakkelijk, we gaan gewoon de nummers 1-100 invoeren, en we doen dit natuurlijk met een for-lus :-)

Voorbeeld foreach3.php -- broncode foreach3.php

```
 <?php
require("header.php");

# De bewaker een nummer laten trekken
$zijnnummer=rand(1, 100);

# Onze array definiïeren
# maximuwaarde = 100
$max=100;

for($teller=1;$teller <= $max;$teller++) {
        $bordjes[$teller]=$teller;
}
foreach($bordjes as $value) {
        if ($value==$zijnnummer) {
                echo "Na $bordjes[$value] pogingen, stemden de nummers
overeen";
        }
}
require("footer.php");
?>
```

In de eerste plaats bepalen we hoeveel bordjes er moeten zijn. We maken nu door middel van een for-lus een array $bordjes aan, en we voegen gewoon de waarde van de $teller toe aan het element van de array. Dus, als de $teller gelijk is aan één dan wordt één toegevoegd aan $bordjes[1], enz ..., totdat we de lus 100 maal hebben doorlopen. Dan lezen we met foreach de aangemaakte array door, en door middel van een simpele if instructie controleren we of de waarde van de array overeenstemt het nummer van de bewaker.

Er zijn er nu die denken waarschijnlijk dat je die waarde ook via een for-lus kunt achterhalen uit die array. Inderdaad, alleen weten wij hier nu uit hoeveel elementen de array bestaat, aangezien we die zelf op 100 hebben geplaatst. Wat als je dit nu wenst te doen met gegevens die je uit een database haalt ? Is het dan niet mogelijk ? Jawel, PHP heeft nog andere instructies om dit te weten te komen, namelijk count.

De funktie count()

Met count(naam_van_de_array) kom je te weten hoeveel elementen in de array aanwezig zijn. In ons geval kunnen we dan plaatsen

```
$elementen=count($bordjes);
```

Voorbeeld count1.php -- broncode count1.php

```
 <?php
require("header.php");
```

```
# De bewaker een nummer laten trekken
$zijnnummer=rand(1, 100);

# Onze array definiïeren
# maximuwaarde = 100
$max=100;

for($teller=1;$teller <= $max;$teller++) {
        $bordjes[$teller]=$teller;
}
$elementen=count($bordjes);
for($teller=1;$teller <= $elementen;$teller++) {
        if ($teller==$zijnnummer) {
                echo "Na $bordjes[$teller] pogingen, stemden de nummers
overeen";
        }
}
require("footer.php");
?>
```

Zoals je ziet gaat het ook op die manier :-).

We hebben nu de lussen gezien en een paar vorbeelden. Ik zal jullie nu nog eens een belangrijke tip meegeven over het gebruik van de foreach lus. Programmeren is progammacode intikken, dat zullen jullie alvast al weten nu. In de code kunnen foutjes zitten, of het resultaat is niet wat je wenste. Je kunt je dan scheel kijken op de code, maar je kunt ook hulpmiddeltjes toepassen. Het echo commando is een grote hulp.

We hebben bij de paswoorden gezien dat we kunnen inloggen met paswoorden, en dat die gegevens worden doorgestuurd naar een andere pagina. We weten inmiddels ook dat PHP hashes maakt van de velden van een formulier. Begint de frank (of moet ik nu euro zeggen ?) al te vallen ? Wat als de gegevens die je verwachtte anders zijn dan je dacht ? Wel we vragen die gegevens nu gewoon op met de foreach lus. We nemen daarvoor terug on script logmetpas2.php dat we kopiëren naar foreach4.php.

Voorbeeld foreach4.php -- broncode foreach4.php

```
 <?php
require("header.php");
?>
<?php
$paswoord=$_POST["paswoord"];
echo "<h1><font color=\"#2208cc\"> Gelieve in te loggen met Uw
paswoord</font></h1>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "<p><input type=\"password\" name=\"paswoord\"></p>\n";
echo "<input type=\"submit\" name=\"versturen\" value=\"Paswoord
versturen\">\n";
echo "</form><br>\n";
if (isset($paswoord)) {
        switch($paswoord) {

        case "1234":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
```

```
        break;

        case "2345":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";

        break;

        case "3456":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";

        break;

        case "4567":
        echo "<h2><font color=\"#db2408\"> Leuk dat je er bent !
</font></h2>\n";
        echo "<form action=\"1234.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leuke madam\">\n";
        echo "</form>\n";
        echo "<form action=\"2345.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"3456.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
```

```
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar een nog leukere madam\">\n";
        echo "</form>\n";
        echo "<form action=\"4567.php\" method=\"post\"\n>";
        echo "<input type=\"hidden\" name=\"paswoord\"
value=\"$paswoord\">\n";
        echo "<input type=\"submit\" name=\"versturen\" value=\"Ik mag
kijken naar de leukste madam\">\n";
        echo "</form>\n";

        break;

        default:
        echo "<h3><font color=\"#db2408\">Leuk geprobeerd, maar ....
</h3>\n";
        echo "<h3><font color=\"#db2408\">Geen toegang zonder geldig
paswoord !</h3>\n";
        }
}
if (isset($versturen)) {
        foreach($_POST as $key => $value) {
                echo "$key=$value<br>\n";
        }
}
?>
<?php
require("footer.php");
?>
```

Ik heb nu in het vet en rood aangeduid wat ik heb toegevoegd aan het script. Als je nu het script runt, zal dat identiek gebeuren als voorheen, maar je zult nu plots extra informatie zien, namelijk de waarden van de variabel $_POST die automatisch door PHP werden toegekend. Laat dergelijke code staan tot je script perfect werkt. Je kunt die uitschakelen door er een "#" voor te plaatsen voor elk lijntje. Mocht er plots iets niet meer werken, haal de "#" terug weg en misschien zie je dan wat er verkeerd gaat.

Als je script perfect werkt, kun je die extra code verwijderen. Dit geldt nu niet enkel voor foreach, maar voor elk script. Zit je in een lus, plaats een echo command die de waarde uitprint van wat je verwacht. Zo is he makkelijker fouten op te sporen.

Nu heb je de basis in handen om PHP code te schrijven. Nu begint de fun. We zullen in de volgende lessen steeds die basis nodig hebben en je zult tevens nog meer ontdekken van de kracht van PHP.

Weg met mailto, leve PHP

Met PHP is het perfect mogelijk om de inhoud van een formulier te versturen via email. Het bericht wordt verstuurd in samenhang met sendmail of een variant ervan op een unix server. We zullen in de eerste plaats leren hoe het versturen van email in elkaar steekt. Daarna zullen we zoals op proffesionele sites, formulieren gaan exploiteren. We gaan een mini-mailer in elkaar knutselen, dan een universele mailer om te eindigen met een maxi-formulier met alle toeters en bellen. Wat hierna volgt, zal enkel en alleen werken als de paginas op een server staan.

De funktie mail()

Voor de gebruikers van Microsoft Windows !

> Ik krijg hier massa's mail van personen die Windows gebruiken die mij vertellen dat het mailen niet lukt. Dit lukt inderdaad niet aangezien Windows standaard geen mailserver aan boord heeft. Dus probeer ergens aan een mailserver te geraken, installeer en configureer die voor je systeem, en het mailen gaat dan wel lukken. iMail express is een gratis e-mail server die je vindt op de site van ipswitch.

Wil je nu een email versturen vanuit een script in PHP ? Niets is eenvoudiger. We gebruiken daarvoor de funktie mail()! De basissyntax ziet er als volgt uit :

```
mail("begunstigde", "onderwerp", "bericht", "From: verzender")
```

Je kunt die waarden rechtstreeks in het script aanbrengen. Het nadeel hiervan is, dat je telkens het script moet aanpassen als de begunstigde iemand anders is. Dus zullen we, wat had je gedacht, variabelen gebruiken. Als de verzending correct verloopt, geeft de funktie mail() de waarde true terug, indien niet, de waarde false.

We gaan nu dadelijk deze kennis toetsen aan een voorbeeld. In het geval dat de mail correct werd verzonden krijgt de bezoeker de melding, Bedankt ! Je mail werd verstuurd. In het geval het versturen niet lukte, krijg je de bijpassende foutmedling.

Voorbeeld mimimailer.php -- broncode minimailer.php

```
<?php
require("header2.php");
echo "<h1>Mini-Mailer</h1><hr>\n";
echo "<p>Verstuur mij een email !</p>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "Uw email adres ? <input type=\"text\" name=\"mail\"><br><br>\n";
echo "Over wat wil jij mij iets laten weten ?<br><br>\n";
echo "<textarea name=\"message\" cols=\"50\" rows=\"5\" wrap=\"soft\">\n";
echo "</textarea><br>\n";
echo "<input type=\"submit\" value=\"Bericht Versturen\">\n";
echo "<form>\n";

if (isset($_POST["mail"]) && $_POST["mail"] != "") {
        if(mail("serge@essetee.be", "You've got mail !", "$message", "From:
$mail")) {
                echo "<p>Bedankt voor jouw mail, jouw bericht werd
verstuurd</p>";
        }
        else {
        echo "<p>Sorry, uw bericht kon niet worden verzonden !</p>";
        }
}
require("footer.php");
?>
```

Je ziet, het script is niet groot. In de eerste plaats checken we met isset of de gebruiker zijn mailadres heeft ingevuld. Indien ja, versturen we de mail. Je ziet dat ik de mail naar mezelf stuur, serge@laptop.essetee.be. Je kunt natuurlijk een To: veld toevoegen en daar dan de waarde van To in plaatsen. Maar pas op hiermee, het doel hiervan is dat jij de mail krijgt,

en niet dat ze deze mailer gaan gebruiken om naar iedereen mails te sturen. Daarom, laat het To field best weg, wil je niet in de problemen komen te zitten dat ze via jouw computer schadelijke mail rondzenden. You've got Mail wordt dan het subject, dan volgt het bericht en van wie het komt. Ik denk dat dit duidelijk is voor iedereen.

Nadat ik op de knop versturen heb geklikt, krijg ik het bericht dat mijn mail werd verstuurd. Dan zie ik op mijn systeem :

```
serge@laptop:~$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/serge": 1 message 1 new
>N  1 serge@laptop.esse  Mon Feb 10 11:50   15/525   You've got mail !
&
```

Dan open ik de mail om hem te lezen, en zie ik wat ik had ingetikt in het tesktvenster.

```
 & 1
Message 1:
From serge@essetee.be Mon Feb 10 11:50:12 2003
Envelope-to: serge@laptop.essetee.edu
To: serge@laptop.essetee.edu
Subject: You've got mail !
From: serge@laptop.essetee.edu
Sender: Serge Terryn
Date: Mon, 10 Feb 2003 11:49:31 +0100

Even testen of het werkt

&
```

En zo te zien werkt het perfect :-)

Jullie kunnen nu die mini-mailer even uittesten. Een bedankwoordje voor deze cursus is graag meegenomen. Maak echter geen misbruik van die mini-mailer, hij staat er om te laten zien dat het werkt. Als je een mailtje hebt verstuurd op mijn server, zal dit bij jouw ook wel werken. Dus probeer het eens. Mocht ik zien dat ik veel stomme mail krijg van mensen die toch de plezanste wensen uit te hangen, zal ik het script zodanig aanpassen dat het niet meer lukt, tot spijt van de anderen die het wel goed menen.

Nu dat mail script is mooi, wat nog beter is, verwijder alle html-code en hou alleen het php-script over en sla het op als mail.php.

Voorbeeld mail.php

```
<?php
require("header2.php");
echo "<br><br><br><br><br><br><br><br>";
$mail=$_POST["mail"];
$message=$_POST["message"];

if (isset($mail) && $mail != "") {
        if(mail("serge@essetee.be", "You've got mail !", "$message", "From:
$mail")) {
        echo "<div align=\"center\"><p><h3>Bedankt voor jouw mail, jouw
bericht werd verstuurd</h3></div></p>";
        }
        else {
        echo "<div align=\"center\"><p><h3>Sorry, uw bericht kon niet
worden verzonden !</h3></div></p>";
        }
}

require("footer.php");
```

```
?>
```

Door alle html te verwijderen houden we juist het script over. Dit script zullen we dan later nog gebruiken. Door gewoon het script aan te roepen, kunnen we mails versturen :-)

Een kleine oefening voor jullie. Maak van onze minimailer een mailclient die je kunt gebruiken om naar iedereen mails te sturen. Ik laat jullie zien hoe het er ongeveer uit moet zien. Je kunt natuurlijk de bron van de pagina bekijken of zo meteen de sourcecode bekijken, maar daar heb je niet veel aan. Probeer het eens zelf.

Voorbeeld : zo zou het eruit moeten zien -- broncode van mijn emailclient.

Zo heb je altijd een mailclient bij de hand om vlug een meiltje te sturen als je aan het surfen bent :-)

De maxi-mailer

Je bent het waarschijnlijk ook al eens tegengekomen. Je verstuurde een mail en dan, deju ik heb daar iets verkeerds ingezet. Wat zou je ervan denken om vooraleer de mail te versturen, je de gebruiker nog eens de kans geeft om alles na te lezen voordat de mail daadwerkelijk wordt verstuurd ? Als de gebruiker een veld vergeet in te vullen moeten we hem daar attent op maken. Wat als er nu twee gebruikers terzelfdertijd een mail willen verzenden ? Onze maxi-mailer zal hier uitweg bieden.

Voorbeeld maximailer.php -- broncode maximailer.php

```
<?php
require("header2.php");
echo "<h1> Een opinie formulier opmaken </h1><hr>";
        if (empty($submit)) {
        // is de submit button gelijk aan 0 of niet gedefinieerd ?
echo "<h3>1. Gelieve Uw gegevens in te vullen</h3>";
echo "<form action=\"$PHP_SELF\" method=\"post\">";
echo "Uw Voornaam : <input type=\"text\" name=\"voornaam\"><br><br>";
echo "Uw Familienaam : <input type=\"text\" name=\"familienaam\"><br><br>";
echo "Email adres : <input type=\"text\" name=\"mail\"><br><br>";
echo "<h3>Wat vindt jij van deze cursus PHP ?</h3><br><br>";
echo "<textarea name=\"opinie\" cols=\"50\" rows=\"5\" wrap=\"soft\">";
echo "</textarea><br><br>";
echo "<input type=\"reset\" name=\"reset\" value=\"Alles wissen\">";
echo "<!-- belangrijk, noem de verzendknop submit ! -->";
echo "<input type=\"submit\" name=\"submit\" value=\"OK - Verzenden !\">";
echo "</form>";
}
else {
//als submit is gedefinieerd
echo "<h3> 2. Gelieve de gegevens na te zien</h3><br>";
foreach($_POST as $key => $value) {
// hier lopen we alle velden van het formulier na
if (empty($value)) {
// is er een veld ledig, nul of niet gedefinieerd ?
echo "<p> Gelieve alle velden in te vullen a.u.b. !</p><br>";
echo "<form>";
echo "<input type=\"button\" value=\"Terug naar vorige pagina\"
onclick=\"javascript:history.back()\">";
echo "</form>";
echo "<!-- Voor hen die geen javascript hebben : -->";
echo "<noscript>Klik op de vorige knop in je browser om terug te keren naar
de vorige pagina.</noscript>";
```

```
exit; // verlaat de lus alsook het programma !
}
}
echo "<p>Dag <strong>$voornaam $familienaam !</strong><br>\n";
echo "<p>Uw email adres is <strong>$mail</strong>!</p>\n";
echo "<p> Jij schreef :</p>\n";
echo "<i>" . stripslashes(nl2br($opinie)) . "</i><br><br>\n";
echo "<p>Zijn de gegevens correct ?</p><br>\n";
$message="$voornaam $familienaam,$mail heeft geschreven\n$opinie";
$message=htmlspecialchars($message);
echo "<form action=\"mail.php\" method=\"post\">\n";
echo "<input type=\"hidden\" name=\"mail\" value=\"$mail\">\n";
echo "<input type=\"hidden\" name=\"message\" value=\"$message\">\n";
echo "<input type=\"button\" value=\"Neen\"";
echo "onclick=\"javascript:history.back()\">\n";
echo "<noscript>Gebruik de terug toets in je browser om naar de vorige
pagina terug te keren !</noscript>\n";
echo "<input type=\"submit\" name=\"mail versturen\" ";
echo "value=\"Alles is correct\"></form>\n";
}

require("footer.php");
?>
```

Waw, een ganse boterham niet ? Zoals je kunt zien gebruiken we hier een beetje van alles wat we tot nu toe al hebben geleerd. Alsook een paar nieuwigheden.

De funktie empty()

Variabelen testen met de funktie empty(). Empty() is het tegenovergestelde van isset(). Isset controleert of een variabele is gedefinieerd, en het is enkel dan dat isset de waarde true terug geeft. Met empty() daarentegen, kun je nagaan of een variabele niet gedefinieerd is, de variabele leeg is of overeenkomt met 0 (nul). In deze gevallen geeft empty() de waarde true door.

Verzoek if en $submit

Het eerste if verzoek in ons script test of er een variabele $submit niet gedefinëerd is. Dit is inderdaad zo, aangezien $submit pas aktief wordt na het versturen van het bericht. Je moet er wel voor zorgen dat de naam van de verzendknop submit is, anders zal dit niet werken.

Het formulier wordt getoond, de gebruiker kan alles invullen en als de gebruiker dan op de knop verzenden klikt, dan pas wordt $submit gedefinïeerd. Na het verzenden wordt de else lus afgewerkt.

Verder zie je ook dat ik regelmatig gebruik maak van javascript. Dit om ervoor te zorgen dat de gegevens niet verloren gaan als je terug moet om iets te corrigeren. Voor het geval javascript niet aanstaat bij de gebruiker, wordt dan vermeld dat hij op de back button van zijn browser moet klikken om terug te gaan naar de vorige pagina.

Met exit verlaat je de lus alsook het programma. Als er een veld niet werd ingevuld, moeten we verhinderen dat de lus opnieuw wordt uitgevoerd alsook moeten we verhinderen dat de javabutton opnieuw wordt getoond. Daarom gebruiken we exit.

Als alle velden correct werden ingevuld, komt het script terecht op de

plaats waar alle informatie die de gebruiker intikte worden getoond met het commando echo.De gebruiker kan de gegevens nog veranderen of aanpassen indien nodig.

We maken dan de variabele $message aan, die alle info van de gebruiker bevat en als we op de knop alles is correct klikken, worden de gegevens doorgegeven naar ons mail.php script die dan zorgt voor de verzending ervan.

Je ziet ook dat ik hier gebruik maak van stripslashes en nl2br zoals in vorige lessen al werd uiteen gedaan.

De gegevens van het formulier worden door middel van input type=hidden doorgegeven aan mail.php.

Nu we een mailer hebben op ons systeem, kunnen we verder gaan met de volgende les. Ik had jullie verwittigd, PHP is niet moeilijk, maar je moet wel bij de pinken blijven. Voor iemand die niets van programmeren afwist, kan ik best begrijpen dat het nu wat moeilijker wordt. Als je echter de vorige lessen goed onder de knie hebt, moet je dit script goed kunnen volgen.

Nu hebben we een opinieformulier aangemaakt, maar je kunt om het even welk formulier aanmaken. Je hoeft enkel de gegevens samen te brengen in de variabele $message en klaar is kees. Je kunt dan de inhoud van om het even welk formulier waarbij je gegevens vraagt van een gebruiker, naar je laten mailen.

Een dergelijk voorbeeld vindt je hier : maak een dhcpd.conf file online aan

En als jullie het netjes vragen, kunnen jullie ook de broncode te zien krijgen.(tip om de minimailer te gebruiken ?) Hier zit wel een addertje onder het gras, je hebt verplichte en niet verplichte velden.

Het volgende addertje onder het gras is de correcte weergave van *option domain-name-server "domein.com";*. Zoals je ziet staan hier aanhalingstekens. De file moet correct getoond worden in html en moet correct verstuurd worden ! Dit is te bereiken via een ommeweg :-).

Tellers, een gastenboek en werken met files.

Dit alles gaan we nu behandelen in deze les. Ik zal jullie leren hoe je files opent, deze leest en terug wegschrijft. Bij gebrek aan het moment van een database, moeten wij ons behelpen met tekstfiles. Je zul dan ook merken dat het gebruik van een database eigenlijk niet echt van doen is onder PHP. Als je veel gegevens hebt te verwerken, dan zul je wel naar een database moeten grijpen.

Een teller installeren

Laten we beginnen met het maken van een simpele teller, die bijhoudt hoeveel maal een webpagina werd bezocht. Om te beginnen maken we een file aan teller.txt met als inhoud 1. Open met je teksteditor een file, tik er 1 in en sla hem op als teller.txt. Je moet wel de file opslaan in de directory van waaruit je je html-pagina's oproept.

Nu gaan we het script schrijven teller.php -- broncode teller.php

```
<?php
require("header.php");
echo "<P><h3>Deze pagina werd ";
$pointer=fopen("teller.txt","r+");
```

```
$teller=fgets($pointer,7);
$teller=trim($teller);
echo "<b>$teller</b> ";
$teller++;
rewind($pointer);
fputs($pointer,$teller);
fclose($pointer);
echo " maal getoond.<h/3></p>";
require("footer.php");
?>
```

Als je nu klikt op teller.php zal de waarde 1 getoond worden. Telkens je nu op de reload button van je browser klikt, zal de teller met 1 verhoogd worden. We gaan nu even het script van naderbij bekijken.

```
$pointer=fopen("teller.txt","r+");
```

Een file openen met fopen()

Om een file te openen heb je de funktie fopen() nodig. De syntax is : $variabele=fopen(naam_van_de_file,mode). Hier is de file geen groot misterie, gewoon de file teller.txt dat we eerst hebben aangemaakt met de waarde 1 erin. Met mode specifieer je wat je kunt doen met de file. De modes zijn :

r	Alleen lezen
r+	Lezen en schrijven
w	Alleen schrijven
a	Append of toevoegen

In ons voorbeeld gaan we de file openen, lezen en dan de waarde verhogen en dan de nieuwe waarde wegschrijven. Dus moet onze file geopend worden voor lezen en schrijven, dus r+. Belangrijke opmerking, in deze mode begint het lezen en schrijven in het begin van de file.

Nu, de variabele $pointer. Je kunt deze variabele om het even welke naam geven, maar gebruik bij voorkeur pointer, want in feite is het een pointer. Als je de funktie fopen() gebruikt, positioneert die funktie een pointer in de tekstfile. Eenmaal de file geopend, blijven we werken met deze pointer. In de mode r+ wordt de pointer dan geplaatst helemaal linksboven in de file. Je moet je hierbij voorstellen dat je tekstfile in een raster wordt verdeeld. Laten we de X-as met letters aanduiden en de Y-as met cijfers.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
a	m	e	t		d	e			
b	t	r	e	i	n				
c	z	o	u		j	e		e	r
d	a	l		z	l	j	n		

Als je nu de funktie fopen() toepast op deze file, zal de pointer op a,1 staan of op de letter m. Doe je fopen($pointer,6), dan zal de pointer op de letter e staan. De pointer blijft daar gewoon staan en wacht op verdere instructies. Welke instructies ? Wel, de volgende bijvoorbeeld :-)

De funktie fgets()

De funktie fgets leest gegevens uit een file. Het lezen wordt beïndigd bij

het einde van de lijn of het einde van de file, of nog, als er een aantal bytes wordt opgegeven. Hier gebruiken we in ons voorbeeld $teller=fgets ($pointer,7). Als we met fopen() onze file teller.txt openen, dan staat de pointer helemaal links boven in onze file op het cijfer 1. Nu zeggen we aan fgets, dat ie de file moet lezen, maar enkel 7 bytes van die file. Dus begint fgets te lezen en schuift 7 bytes door naar rechts in onze file. Aangezien er thans maar 1 cijfer instaat zijn de rest blanco bytes. Mocht er nu het getal 123456789 instaan, dan zal fgets stoppen met lezen op het getal 7, alles wat erna komt zal genegeerd worden.

Vervolgens kennen we de waarde dat fgets retourneert toe aan onze variabele $teller, en verhogen we onze teller met 1 (teller++). Ben je nog mee ? Zoals gezegd heeft fgets nu 7 bytes gelezen van onze file. Dus, onze onzichtbare cursor, de pointer staat nu op het eerste lijntje van onze file, zeven bytes naar rechts.

De funktie trim()

De funktie trim() gebruiken we om spaties te verwijderen aan het begin en of aan het einde van een string. Als we een tekstfile aanmaken hebben we nogal eens de neiging van op enter te drukken na een invoer. Op enter drukken betekent echter dat er in je tekstfile een onzichtbare einde regel wordt ingevoegd. Nu lezen we een getal in, en als je op enter hebt gedrukt toen je het getal 1 invoerde in de tekstfile, zit je daar dan met een einde regelteken. Aangezien we gaan rekenen met dat getal, moeten we enkel de cijfers hebben, want herinner de eerste lessen, als je strings met elkaar gaat optellen kun je onverwachte resultaten krijgen. Om nu die blank spaces of einde regeltekens te verwijderen gebruiken we de trim funktie. Met trim kun je allerhande tekens verwijderen, sla er de php-manual op na als je er meer wenst over te weten. Onthou alleen, dat met de funktie trim, zonder speciale argumenten, de spaties en einde regeltekens worden verwijderd. Voor ons voorbeeld is dat voldoende.

De funktie rewind()

Als we nu gaan schrijven in onze file, moeten we terug aan het begin van onze file zijn. Met rewind($pointer), plaatsen we onze cursor nu terug links boven in onze file. Rewind kan ook het byte argument meekrijgen. rewind($pointer,2), zal onze pointer bijvoorbeeld van op zijn huidige positie twee bytes naar links laten terugkeren. Zonder bijkomend argument wordt de cursor op a,1 geplaatst.

De funktie fputs()

Met deze funktie gaan we tenslotte schrijven in onze file. fputs ($pointer,$teller) vertelt dat hij moet beginnen schrijven op de plaats waar onze onzichtbare cursor staat en dat hij de waarde van $teller moet wegschrijven. Nu, zolang je teller onder de 7 cijfers blijft, zul je geen problemen ondervinden. Mocht je teller nu groter worden, dan zul je het script moeten aanpassen, want zoals gezegd, fgets stopt met lezen op positie 7 in ons geval, en alles wat erna komt wordt genegeerd.

De funktie flcose()

Deze funktie is klaar en duidelijk me dunkt. Met fclose(teller.txt) sluiten we onze file terug af op een nette manier. Let wel, het is niet verplicht om fclose te gebruiken, zonder fclose zal het ook werken, maar door je files netjes af te sluiten met fclose, laadt je de file uit het geheugen en spaar je alzo systeem resources. Dus, niet vergeten, sluit je files af met fclose.

PERMISSIES !!!!!!

Nu, mensen die met Windows werken zullen daar geen problemen mee hebben. Zoals we weten kan iedereen op een Windows systeem zomaar files aanmaken, verwijderen en overschrijven. Onder Unix ligt dit wel anders. Je webserver wordt opgestart als user apache of www-data of misschien onder nog een andere user. Als je de standaard apache root directory gebruikt, is de eigenaar van de files root, dit wil zeggen dat enkel en alleen root naar een file zal kunnen schrijven en in het ergste geval, enkel root een file kan lezen. Dus, de file moet world-writeable zijn, zoniet, zul je permissions denied krijgen als je wilt schrijven als user apache naar een file. Doe dus een chmod 666 op de file teller.txt om deze fouten te voorkomen.

Nu, met onze kennis van het lezen en schrijven naar files, is het moment aangebroken om ons gastenboek online te plaatsen. Wat doet een gastenboek ? Heel eenvoudig, de mensen kunnen iets schrijven erin, dat anderen op hun beurt terug kunnen lezen. Het is van belang, dat de laatste opmerkingen steeds bovenaan staan. Dus, zullen we de file moeten inlezen, de nieuwe tekst bovenaan toevoegen en het geheel terug wegschrijven. Klaar ? Hier gaan we dan.

Ons gastenboek gastenboek.php -- broncode gastenboek.php

Maak nu eerst een lege file commentaar.txt aan en voor de unixers, doe een chmod 666 op die file.

```
<?php
require("header.php");
echo "<H2>Een eenvoudig gastenboek</H2>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "<p>Uw naam : <input type=\"text\" name=\"naam\"></p>\n";
echo "<p>Uw E-mail : <input type=\"text\" name=\"email\"></p>\n";
echo "<p>Uw commentaar :</p>\n";
echo "<textarea cols=\"55\" rows=\"5\" wrap=\"soft\"
name=\"commentaar\"></textarea><br>\n";
echo "<p><input type=\"submit\" value=\"Commentaar verzenden\"></p>\n";
echo "<hr>\n";
echo "<p><h2>Reeds geleverde commentaar</h2></p>\n";

$tekstfile="commentaar.txt";
if (isset($commentaar) && naam != "" && email != "") {
        $pointer=fopen($tekstfile,"r+");
        $oudcommentaar=fread($pointer,filesize($tekstfile));
        $email="<a href=\"mailto:$email\">$email</a>";
        $date=date("j.n.Y");
        $commentaar=htmlspecialchars($commentaar);
        $commentaar=stripslashes(nl2br($commentaar));
        $geschreven="<p><strong>$naam</strong> ($email) heeft geschreven op
<i>$date</i> :
                    <br><br>$commentaar</p><hr>\n";
        rewind($pointer);
        fputs($pointer,"$geschreven\n$oudcommentaar\n");
}
readfile($tekstfile);
require("footer.php");
?>
```

We hebben hier nu nog een paar nieuwe funtkies gebruikt. Ik leg deze hierbij uit.

De funktie fread()

fread() doet het tegenovergestelde van fgets(). fgets() werkt lijn voor lijn, fread() leest ineens de inhoud van een ganse file in. fread neemt ook argumenten aan. fread(pointer,bytes). Maar aangezien we op voorhand niet weten hoe groot de file zal zijn, die wordt natuurlijk steeds maar groter naarmate er meer commentaar wordt geleverd, hebben we iets anders bij de hand die dit voor ons oplost.

De funktie filesize()

De funktie filesize() retourneert het aantal bytes in een file. Hier neemt de variabel $tekstfile de waarde van commentaar.txt aan. Ik moet nu die file inlezen als oud commentaar. Ik doe dit dan als volgt : $oudcommentaar=fread($pointer,filesize($tekstfile));. De filesize in dit geval wordt dan de volledige inhoud van $tekstfile of van onze commentaar.txt file.

$email="$email"; Hier maak ik een Email link aan.

De funktie date()

Ik voeg met de funktie date de datum in wanneer iemand commentaar levert. Date heeft een massa argumenten die ik hier voor jullie netjes op een rij ga plaatsen.

Wat de datums aangaat :

Argument	Uitleg	Voorbeeld
j	dagen van de maand niet voorafgegaan door een 0	van 1 tot 31
d	dagen van de maand voorafgegaan door een 0	van 01 tot 31
n	maanden niet voorafgegaan door een 0	van 1 tot 12
m	maanden voorafgegaan door een 0	van 01 tot 12
S	Engelse notatie	1st 3rd 5th
y	Jaargetal in 2 cijfers	03
Y	Jaargetal in 4 cijfers	2003
D	Dagen van de week verkort	mon tue
L	Dagen van de week lange schrijfwijze	monday tuesday
M	Maanden van het jaar kort	jan feb mar
F	Maanden van het jaar lang	januari
w	dag van de week (zondag=0)	van 0 tot 6
z	dag van het jaar	van 0 tot 365
t	hoeveel dagen een maand bevat	van 28 tot 31

Wil je nu de datum schrijven als 24.02.2003 dan gebruik je date("j.n.Y") of bijvoorbeeld de 4th of July 2003 dan gebruik je date("jS of F Y").

Wat de uren aangaat :

Argument	Uitleg	Voorbeeld
a	am of pm	am
A	AM of PM	PM
h	uren in 12 uur formaat	van 1 tot 12
H	24 uren formaat	van 0 tot 24
i	minuten	van 0 tot 59
s	seconden	van 0 tot 59

Met date("H:i:s") wordt de huidige tijd getoond bij het opvragen van de pagina.

De funktie readfile()

Met readfile(file_naam) kun je een ganse file ineens inlezen en laten weergeven in je browser. Je ziet in mijn script dat ik alles wat door de gebruiker geschreven wordt opgenomen wordt in de variabele $geschreven. Als ik mijn file dan wens te schrijven, doe ik een rewind van de $pointer en schrijf ik eerst $geschreven weg, gevolgd door $oudcommentaar, zijnde de inhoud van de file voordat de nieuwe gebruiker zijn woordje heeft gedaan. Zo zal het laatste commentaar steeds bovenaan staan. Wil jij het liever onderaan ? Geen probleem, schrijf dan eerst oudcommentaar weg gevolgd door geschreven.

Tot zover het werken met files. We zijn klaar voor de volgende les, de cookies. In deze les zal ik jullie dan nog een handige funktie tonen voor het gebruik met files. Remember dat ik bij de paswoorden heb gezegd dat je die ook in een file kunt plaatsen ? Wel in de volgende les zullen we dan ook zien hoe je alles kunt uitlezen met een handige funktie.

De cookies

"Blij je terug te zien ..... ". Op hoeveel webpaginas heb je dit al gezien of iets dergelijks ? En het ergste van al, het is permanent. Weken nadat je een online shop bezocht hebt bijvoorbeeld, die weet nog steeds wat er in je winkelmandje zat bij je vorig bezoek. Wat of wie verbergt zich daarachter ? Magie ? Big Brother is watching You ? Helemaal niet dus. Het zijn eenvoudige cookies die aan de basis van dit alles liggen.

Wat zijn cookies ?

Cookies zijn één van de meest mysterieuse en onverstaanbare technieken op het web. Sommigen beweren dat cookies kunnen dienen om je harde schijf te controleren of dat cookies een open poort zijn voor virussen. Niets van dit alles is dus waar. Cookies zijn niets anders dan gewone tekstbestanden die een webpagina toelaten gegevens te registreren op jouw computer.

Cookies zijn niet gevaarlijk, zoals gezegd, zijn het gewone tekstbestandjes waarin je webbrowser gegevens kan schrijven. Als je terug naar dezelfde webpagina surft waar het cookie vandaan komt, kan je webbrowser terug dit bestandje inlezen en eventueel de nodige gegevens verstrekken die de webpagina vraagt, zonder dat jij iets hoeft in te tikken.

Normaal gezien weet de exploitant van de webpagina niet dat hij een cookie op jouw computer plaatst. Nu zijn er wel exploitanten van webpaginas die deze cookies gebruiken met eventuele andere informatie die jij verstrekt. Op die manier kunnen ze het surfgedrag van een gebruiker bepalen, en zo de gewenste paginas waarin de surfer interesse bleek te hebben, direkt voorschotelen bij een volgend bezoek. Cookies worden enkel gevaarlijk als je daarin je telefoonnummer, je adres of elektronisch adres in achterlaat. Cookies kunnen dienen om je later te spammen.

Webbrowsers zijn echter goed in staat om bepaalde cookies van een bepaald domein te verbieden. Hoe je dat doet, hangt af van de gebruikte webbrowser. Als je onder Mozilla bjvoorbeeld klikt op edit->preferences->Privacy & Security->cookies en dan klikt op Manage Stored Cookies, krijg je heel wat info te zien over de cookies op jouw systeem. Klik op een cookie om de inhoud ervan te zien.

Cookies plaatsen onder PHP

Cookies onder PHP worden geplaatst door middel van de funktie setcookie(). Deze funktie kent verschillende argumenten. Verplichte zijn, de naam van de cookie en de waarde wan de cookie. De funktie gebruiken we dus als volgt setcookie("naam_cookie","waarde_cookie"). Let er wel op, cookies zijn strings.

Laten we nu even als voobeeld nemen : je stuurt een cookie naar een webbrowser die moet onthouden of de bezoeker als eens langs is geweest of niet. We noemen ons cookie dan ook bezoek en de waarde zetten we op ja. We schrijven dit dan zo :

```
setcookie("bezoek","ja");
```

BELANGRIJK !

Je kunt een cookie niet om het even waar plaatsen in je script. Een cookie staat helemaal bovenaan in je webpagina, en voor de tags. Dus, als je cookies wenst te gebruiken, eerst de cookie plaatsen, dan beginnen aan je webpagina.

Ons eertse cookie : voorbeeld cookie.php -- broncode cookie.php

```
<?php
setcookie("bezoek","ja");
require("header.php");
echo "<h1>Een bezoeker herkennen</h2>";
if (isset($bezoek) && $bezoek=="ja") {
        echo "Wij zijn blij om je <strong> terug </strong> te zien !";
}
else {
        echo "Wij zijn blij je hier te mogen zien !";
}
require("footer.php");
?>
```

Als je nu dit script runt, wordt je de eerste maal begroet met "Wij zijn blij je hier te mogen zien !". Als je nu op de reload button van je browser klikt moet je dan echter "Wij zijn blij om je **terug** te zien !" te zien krijgen. Onze cookie werkt dus.

PHP gebruikt hashes voor het beheer van cookies. De waarden worden in een hash geplaatst $HTTP_COOKIE_VARS. Onze cookie bezoek is dus leesbaar uit $bezoek maar ook uit $HTTP_COOKIE_VARS["bezoek"]. De lijn :

```
if (isset($bezoek) && $bezoek=="ja") {
        echo "Wij zijn blij om je <strong> terug </strong> te zien !";
}
```

Kunnen we ook schrijven als :

```
if (isset($HTTP_COOKIE_VARS["bezoek"]) && $HTTP_COOKIE_VARS["bezoek"]
=="ja") {
        echo "Wij zijn blij om je <strong> terug </strong> te zien !";
}
```

Waarom zo moeilijk ? Dit om veilgiheidsredenen. De tweede methode is de aangeraden methode om te gebruiken. Als je nu formlieren en cookies gaat gebruiken, bestaat er kans dat je je weg niet meer terug vindt in de gekozen namen voor de variabelen. Nog iets, als je global_variables op off staan in php.ini, zul je niet anders kunnen dan de tweede schrijfwijze te gebruiken. Ons script moet er eigenlijk als volgt uitzien :

Voorbeeld cookie2.php -- broncode cookie2.php

```
<?php
setcookie("bezoek","ja");
require("header.php");
echo "<h1>Een bezoeker herkennen</h2>";
if (isset($HTTP_COOKIE_VARS["bezoek"]) && $HTTP_COOKIE_VARS["bezoek"]
=="ja") {
        echo "Wij zijn blij om je <strong> terug </strong> te zien !";
}
else {
        echo "Wij zijn blij je hier te mogen zien !";
}
require("footer.php");
?>
```

Als je nu je browser afsluit en opnieuw opstart en op de link cookie2.php klikt, dan zul je terug verwelkomd worden door de pagina zoals je er nog nooit geweest bent. Lijdt je webbrowser aan Altzheimer of heeft hij ergens geheugenverlies ? Niet van dit alles natuurlijk. Cookies zijn beperkt in hun tijd. Zoals in ons script hebben wij geen levensduurte meegegeven aan onze cookie. Bijgevolg, als je de browser sluit, verwijderd je browser de cookie.

De tijdsduur van een cookie

Voor de tijdsduur van een cookie hebben ze het wat ver gezocht volgens mij. Een minuut = 60 seconden, een uur = 3600 seconden en een dag van 24 uur = 86400 seconden. Met die gegevens geven we onze cookie een tijdsduur onder de volgende syntax :

```
setcookie("naam_cookie","waarde_cookie",vervaldatum)
```

Wil je nu een cookie plaatsen met een tijdsduur van :

- 1 uur is de syntax setcookie("bezoek","ja",time()+3600)
- 3 uur is de syntax setcookie("bezoek","ja",time()+3600*3)
- 1 dag is de syntax setcookie("bezoek","ja",time()+86400)
- 3 dagen is de syntax setcookie("bezoek","ja",time()+86400*3)

Om nu ons script aan te passen en onze cookie een levensduur van 5 uur te geven doen we dit als volgt :

voorbeeld cookie3.php -- broncode cookie3.php

```
<?php
setcookie("bezoek","ja",time()+3600*5);
require("header.php");
echo "<h1>Een bezoeker herkennen</h2>";
if (isset($HTTP_COOKIE_VARS["bezoek"]) && $HTTP_COOKIE_VARS["bezoek"]
=="ja") {
        echo "Wij zijn blij om je <strong> terug </strong> te zien !";
}
else {
        echo "Wij zijn blij je hier te mogen zien !";
}
require("footer.php");
?>
```

Als je nu terug met mozilla je cookies gaat bekijken, zul je nu kunnen vast stellen dat er nu een tijd wordt vermeld bij ons cookie wanneer die vervalt.

Gewapend nu met de kennis van cookies en de tijd, kunnen we nu een poll plaatsen op onze site. We zorgen er met ons cookie voor, dat de bezoeker per dag van 24 uur maar één keer kan stemmen. Als hij terug wil stemmen zal hij dus over 24 uur moeten terug komen. Nu, dit systeem is niet waterdicht, de gebruiker kan de cookies verwijderen, een andere browser gebruiken enz... Dit is allemaal op te vangen, maar dit is kost voor andere lessen. We houden het hier simpel :-)

We gaan nu aan de bezoekers van de site de mogelijkheid geven zich uit te spreken over de moeilijkheidsgraad van de cursus.

1. Zeer eenvoudig
2. Goed verstaanbaar
3. Ik begrijp er niets van

Als de bezoekers hebben gestemd, kunnen zij natuurlijk ook de uitslag van de poll bekijken, die we een gaan voorstellen in de vorm van een diagram.

Maak nu eerst een file result.txt aan met de inhoud 0,0,0 (voor unixers : chmod 666 result.txt)

Voorbeeld poll.php -- broncode poll.php

```
<?php
if (isset($submit)) {
        setcookie("gestemd","ja",time()+86400);
}
require("header.php");
echo "<h1> Onze poll </h1>\n";
echo "<h3> Wat vindt je van deze cursus PHP ? </h3>\n";
echo "<form action=\"$PHP_SELF\" method=\"post\">\n";
echo "<input type=\"radio\" name=\"antwoord\" value=\"0\">\n";
echo "Zeer eenvoudig<br>\n";
echo "<input type=\"radio\" name=\"antwoord\" value=\"1\">\n";
echo "Goed verstaanbaar<br>\n";
echo "<input type=\"radio\" name=\"antwoord\" value=\"2\">\n";
echo "Ik begrijp er niets van<br><br>\n";
if (empty($submit) && empty($gestemd) && empty($antwoord)) {
        echo "<input name=\"submit\" type=\"submit\" value=\"Stem
uitbrengen\"\n";
}
else {
        echo "<p> Bedankt voor jouw stem !</p>\n";
                if (isset($submit) && isset($antwoord) && empty
($HTTP_COOKIE_VARS["gestemd"])) {
                $result="result.txt";
```

```
                $pointer=fopen($result,"r+");
                $poll=fread($pointer,filesize($result));
                $poll=trim($poll);
                $poll=explode(",",$poll);
                $poll[$antwoord]++;
                $poll=$poll[0] . "," . $poll[1] . "," . $poll[2];
                rewind($pointer);
                fputs($pointer,$poll);
                fclose($pointer);
                }
}
echo "</form>\n";
echo "<p>[ <a href=\"polluitslag.php\" target=\"_blank\">Bekijk de
resultaten van de poll</a>]</p><br>\n";
require("footer.php");
?>
```

Uitleg over het script :

De eerste lijn : we gaan nu enkel en alleen een cookie plaatsen als de gebruiker op de submit knop geklikt heeft. We noemen de cookie "gestemd" en geven hem de waarde "ja" van zodra de gebruiker op de submit knop heeft gedrukt om zijn stem uit te brengen.

Het formulier : zijn gewone radio buttons die we de naam antwoord geven. De waarden zijn respectievelijk 0,1 en 2.

De empty variabele : we kijken of de gebruiker al op de submit knop heeft gedrukt en we kijken of er al een cookie bestaat dat ie al gestemd heeft en dat er geen enkel radiobutton is aangevinkt. Indien aan al die voorwaarden voldaan wordt (&&) tonen we de submit knop.

Indien één van die variabelen niet leeg is, betekent dit dat de gebruiker al gestemd heeft, en dan tonen we gewoon de tekst dat hij bedankt wordt voor zijn uitgebrachte stem.

Dan gaan we gewoon de aangemaakte file result.txt inlezen.

De funktie explode()

Hier zien we nu een nieuwe en heel handige funktie van PHP. We hebben nu de file result.txt ingelezen en die waarde toegekend aan $poll. De variabele $poll bevat dus de waarde "0,0,0". Met de funktie explode, gaan we nu die gegevens inlezen in een hash. Je ziet dat de getallen worden gescheiden door een komma. Explode neem als argument aan, het scheidingsteken en de variabele. In ons geval dus een komma. explode(",",$poll). Hadden we nu de file result.txt geschreven als 0%0%0 dan hadden we explode gebruikt als $poll=explode("%",$poll).

De funktie explode leest dus de variabele $poll in, van zodra hij een komma tegenkomt, plaats hij de waarde in $poll[0], leest terug verder tot hij terug een komma tegenkomt, plaats dan de waarde in $poll[1] enz... Als hij aan het einde van de file komt plaats hij dan de laatste waarde in $poll [2].

$poll[$antwoord]++ : als de gebruiker een radiobutton heeft aangevinkt wordt de waarde die aan de radiobutton werd toegekend met 1 verhoogd in de hash. Zoals gezegd, maakt PHP auto hashes van je formulier. Dus voor de eerste optie is dat antwoord[0] dan antwoord[1] enz..

Als de waarde verhoogd werd naargelang de uitgebrachte stem, moeten we de gegevens terug wegschrijven naar result.txt. We gebruiken hier terug de concatenation operator om onze gegevens weg te schrijven. $poll[0] . ",".

We vergeten dus niet om ook de kommas terug op de goede plaats weg te schrijven. Na het wegschrijven als de gebruiker bijvoorbeeld voor optie 1 had gekozen, zal onze file result.txt er dan als "1,0,0" uitzien.

We hebben onze cookie een levensduurte gegeven van 24 uur, dus, als je nu terug de pagina herlaadt, zul je geen submit button meer te zien krijgen en kun je bijgevolg ook niet meer stemmen. Na 24 uur zal de cookie vervallen, en kun je dan opnieuw stemmen :-)

De uitslag van de poll : polluitslag.php -- broncode polluitslag.php

```
<?php
require("header.php");
echo "<h2>Uitslag van de poll</h2>\n";
$result="result.txt";
$pointer=fopen($result,"r");
$poll=fread($pointer,filesize($result));
fclose($pointer);
$poll=explode(",",$poll);
$totaal=$poll[0]+$poll[1]+$poll[2];
$barlengte=400;
$barlengte0=$poll[0]*$barlengte/$totaal;
$barlengte1=$poll[1]*$barlengte/$totaal;
$barlengte2=$poll[2]*$barlengte/$totaal;
$barlengte0=round($barlengte0);
$barlengte1=round($barlengte1);
$barlengte2=round($barlengte2);
## test # echo "$barlengte0 $barlengte1 $barlengte2 de verschillende
lengten<br>\n";
echo "<h2><i>$totaal personen hebben reeds gestemd</i></h2>";
echo "<br>\n";
echo "<table>\n";
echo "<tr><td>Optie 1 = </td><td> Zeer eenvoudig</td></tr>\n";
echo "<tr><td>Optie 2 = </td><td> Goed verstaanbaar</td></tr>\n";
echo "<tr><td>Optie 3 = </td><td> Ik begrijp er niets van</td></tr>\n";
echo "</table><br><br>\n";
echo "<table border=\"0\">\n";
echo "<tr>";
echo "<td><strong>Optie 1</strong></td>\n";
echo "<td> </td><td width=\"$barlengte0\"
bgcolor=\"red\"> </td><td> <i>$poll[0]</td></tr>\n";
echo "</table>\n";
echo "<table border=\"0\">\n";
echo "<tr>";
echo "<td><strong>Optie 2</strong></td>\n";
echo "<td> </td><td width=\"$barlengte1\"
bgcolor=\"green\"> </td><td> <i>$poll[1]</td></tr>\n";
echo "</table>\n";
echo "<table border=\"0\">\n";
echo "<tr>";
echo "<td><strong>Optie 3</strong></td>\n";
echo "<td> </td><td width=\"$barlengte2\"
bgcolor=\"blue\"> </td><td> <i>$poll[2]</td></tr>\n";
echo "</table>\n";
echo "<br><div align=\"center\"><a href=\"../les10.html\">Terug naar de
les</a></div>\n";
require("footer.php");
?>
```

We moeten hier niet schrijven naar de file, dus openen met "r" volstaat hier. Met explode halen we terug onze waarden uit de file result.txt en kennen die toe aan $poll. We tellen nu de waarden tesamen zodat we het

totaal krijgen van de mensen die al hebben gestemd, en plaatsen dit in de variabele $totaal.

Dan de diagram. We zorgen ervoor dat de diagrammen geen afmetingen aannemen dat de gebruiker moet scrollen met zijn browser naar rechts om alles te zien te krijgen. Ik definieer hier de maximumlengte op 400 pixels ($barlengte=400). Nu moeten we die lengten nog proportioneel weergeven. We doen dit door middel van onzichtbare tabellen. Om te voorkomen dat een staaf te lang wordt, delen we de barlengte(400) door het aantal uitgebrachte stemmen per categorie. We ronden de waarden af naar gehele waarden.

De waarden dat we hier verkrijgen gebruiken we om de breedte van de tabel te bepalen en vullen die met een lege spatie en geven we een achtergrondkleurtje mee. Na de staaf shrijven we dan nog eens het aantal stemmen in cijfers.

Dit is dus een vrij simpel script, je kunt dat eventueel zelf uitbreiden en aanpassen.

Zoals gezegd in de vorige lessen, nu we de funktie explode hebben leren kennen, kunnen we nu ook op heel eenvoudige wijze paswoorden uit een file lezen.

Maak nu een file pass.txt aan met volgende inhoud :

```
conny,7891,
serge,linux,
jan,debian,
dirk,3456,
ann,4567,
joni,5678,
bram,6789
```

voorbeeld passfile.php -- brondoce passfile.php

```
<?php
require("header.php");
echo "<h2>Paswoorden uit een bestand lezen</h2>\n";
echo "<form action=\"pascontrole.php\" method=\"post\"\n>";
echo "<p>Tik uw naam in : <input type=\"text\" name=\"naam\"></p>\n";
echo "<p>Tik uw paswoord in : <input type=\"password\"
name=\"pas\"></p>\n";
echo "<p><input type=\"submit\" value=\"Paswoord versturen\"></p>\n";
echo "</form><br><hr>";
require("footer.php");
?>
```

Dit is dus een gewoon formulier om een paswoord in te geven. Na op Paswoorden versturen te hebben geklikt kom je op passcontrole.php terecht.

Voorbeeld pascontrole.php -- broncode pascontrole.php

```
<?php
require("header.php");
$file="pass.txt";
$naam=$_POST["naam"];
$pas=$_POST["pas"];
if (isset($naam) && $naam != "" && $pas != "") {
        $pointer=fopen($file,"r+");
        $paswoorden=fread($pointer,filesize($file));
        $paswoorden=explode(",",$paswoorden);
        $tel=count($paswoorden);
```

```
        for ($i=0;$i<$tel;$i++) {
                $filenaam=$paswoorden[$i];
                $filenaam=trim($filenaam);
                $i++;
                $filepass=$paswoorden[$i];
                $filepass=trim($filepass);
                if (($filenaam == $naam) && ($filepass == $pas)) {
                                echo "<h2>correct paswoord !</h2><br>\n";
                                $wachtwoord="oke";
                }
        }
        fclose($pointer);
}
if (isset($wachtwoord) && $wachtwoord == "oke") {
        }
else {
        echo "<h2>Jammer, maar helaas ... zonder paswoord geen
toegang</h2><br>\n";
        }
require("footer.php");
?>
```

Het script spreekt voor zichzelf. Als het paswoord correct is, kun je dan onder "if (isset($wachtwoord) && $wachtwoord == "oke") { " toelaten wat je wilt.

So folks, de cursus PHP zit erop, ik hoop dat jullie er iets van geleerd hebben. Het heeft mij veel tijd en energie gekost om deze op te stellen, maar ik deed het graag, aangezien ik ook overal info heb moeten zoeken over programmeren met PHP. Met deze basis kun je nu steeds je kennis uitbreiden.

Nog meer PHP stuf :-)

Deze sectie is bedoeld voor diegene onder jullie die de eerste 10 lessen hebben gevolgd. Ik zal hier niet meer in detail gaan, de php code kun je zien, en de code zal voor jullie dan geen probs mogen opleveren.

Een andere manier om variabelen te recuperen

We hebben gezien in de cursus dat je met de register_globals op Off, nu steeds de de volgende syntax moet gebruiken om variabelen te recuperen uit een vorig formulier :

```
$waarde=$_POST[waarde];
```

We hebben nu een handige funktie van php, namelijk import_request_variables ("arg1","arg2") die ons toelaat op een andere en gemakkelijkere manier de variabelen van GET/POST/COOKIE in onze nieuwe pagina te verwerken.

Naargelang welke variabelen we willen importeren gaat "arg1" de waarde g (GET),p(POST),c(COOKIE) krijgen. De schrijfwijze is niet case sensitive, je mag dus zowel gPc als GpC schrijven, het maakt niets uit. Let wel, deze funktie is enkel geldig voor deze soort variabelen. De volgorde van de letters speelt WEL een rol. Als je gpc specifieert, dan zullen de cookie variabelen deze van get en post overschrijven als ze dezelfde naam hebben !

Ons tweede argument "arg2" bevat een prefix waarmee we onze variabelen willen aanspreken. Laten we nu deze waarde even op my_ plaatsen, dan zal ik mijn variabelen kunnen lezen uit $my_waarde.

Onze funktie wordt dan : import_request_variables("gpc","my_")

Een voorbeeld

Laten we aannemen dat we in onze pagina een cookie hebben gespecifieerd met als naam "koekje" met de waarde "lekker". Verder hebben we een formulier gemaakt waarin we een gebruikersnaam ($naam) en een paswoord ($paswoord) hebben gevraagd, en we versturen dit alles door middel van een submitknop ($versturen) en de "POST" methode naar een formulier import2.php.

Ik wil nu natuurlijk in import2.php al deze waarden recuperen. We doen dat dus zo :

```
import_request_variables("pc","my_");
```

Ik kan nu de waarden uitlezen met :

```
$my_koekje
$my_naam
$my_paswoord
$my_versturen
```

We gieten dit alles eens in een pagina : import.php -- broncode import.php

```
<?php
setcookie("koekje","lekker");
include("header2.php");
?>
<form action="import2.php" method="POST">
<table>
<tr><td><input type="text" name="naam" size="25"></td></tr>
<tr><td><input type="password" name="paswoord" size="25"></td></tr>
<tr><td><input name="versturen" value="versturen" type="submit"></td></tr>
</table>
</form>
<?php
include("footer.php");
?>
```

Onze import2.php pagina -- broncode import2.php

```
<?php
include("header2.php");
import_request_variables("pc","my_");
echo "<h4> We gaan nu onze variabelen hier op import2.php
uitlezen</h4><br>\n";
echo "Onze variabele koekje = \$my_koekje : $my_koekje<br>\n";
echo "Onze variabele naam = \$my_naam : $my_naam<br>\n";
echo "Onze variabele paswoord = \$my_paswoord : $my_paswoord<br>\n";
echo "Onze variabele versturen = \$my_versturen : $my_versturen<br>\n";
include("footer.php");
?>
```

Zoals je nu kunt merken, kunnen we nu al onze variabelen die we specifieerden in ons formulier gewoon uitlezen door er $my_ voor te plaatsen, zonder dat we dan al onze variabelen uit de $_POST waarden dienen te herdefinïeren.

Op de oude methode gingen we het als volgt moeten doen :

```
$naam=$_POST[naam];
$paswoord=$POST[paswoord];
```

```
$versturen=$_POST[versturen];
$koekje=$HTTP_COOKIE_VARS["koekje"];
```

De funktie import_request_variables() laat je dus terug toe om op eenvoudige wijze je variabelen te verwerken.

Nog meer met files

We hebben reeds gezien hoe we gegevens uit een file inlezen en de output ervan opvangen. Ik ga hier nu nog enkele voorbeelden geven met verschillende mogelijkheden van fgets().

Zonder de grootte te kennen van een file, gaan we nu even proberen lijn voor lijn te lezen van een tekstfile. We gaan daarbij gebruik maken een while lus die kijkt wanneer wij op het einde van de file zijn gekomen. Wanneer is een file aan zijn einde ? Wel, zowel onder dos of unix hebben we daar een speciale info voor, namelijk eof (end of file). Wanneer het einde van de file bereikt is, wordt een eof weergegeven.

Iedereen ken het gedichtje : Jantje zag eens pruimen hangen ? Wel we gaan dit gedichtje gebruiken voor ons voorbeeld :

```
Jantje zag eens Pruimen hangen

Jantje zag eens pruimen hangen
o, als eieren zoo groot
't scheen dat Jantje wou gaan plukken
schoon zijn vader 't hem verbood
hier is, zei hij, noch mijn vader
noch de tuinman die het ziet
aan een boom, zoo vol geladen
mist men vijf, zes pruimen niet
maar ik wil gehoorzaam wezen
en niet plukken, ik loop heen
zou ik om een handvol pruimen
ongehoorzaam wezen? Neen!

voort ging Jantje naar zijn vader
die hem stil beluisterd had
kwam hem in 't loopen tegen
vooraan op het middenpad
kom mijn jantje, zei de vader
kom mijn kleine hartedief
nu zal ik pruimen plukken
nu heeft vader Jantje lief
daarop ging vader aan het schudden
jantje raapte schielijks op
jantje kreeg zijn hoed vol pruimen
en liep heen op een galop
```

Nu gaan we dit gedichtje eens lijn voor lijn laten uitprinten met het jantje.php script -- broncode jantje.php :

```
<?php
require("header.php");
$mijn_file = fopen ("jantje.txt", "r");
while (!feof ($mijn_file)) {
    $file_inhoud = fgets($mijn_file);
    echo "$file_inhoud<br>\n";
}
fclose ($mijn_file);
require("footer.php");
```

```
?>
```
Zoals je kunt zien, is er nu weinig code nodig om ons gedichtje te laten uitprinten. De while lus loopt steeds onze file door tot het einde bereikt is. Iedere regel wordt uitgelezen met het fgets commando en daarna uitgeprint.

Met de vorige kennis, gaan we nu eens de funktie explode() ook gebruiken op een paswoord file met de volgende inhoud :

```
serge,serge1234
bram,bram1234
joni,j001t
conny,cn007nc
```

Voorbeeld mijnpass.php -- broncode mijnpass.php

```
<?php
require("header.php");
$mijn_file = fopen ("mijnpass.txt", "r");
while (!feof ($mijn_file)) {
    $file_inhoud = fgets($mijn_file, 4096);
    $file_inhoud = explode(",",$file_inhoud);
    $tel = count($file_inhoud);
    for ($i=0;$i<$tel;$i++) {
        $file_inhoud[$i]=trim($file_inhoud[$i]);
        echo "$file_inhoud[$i]<br>\n";
    }
}
fclose ($mijn_file);
require("footer.php");
?>
```

Als je nu in een formulier een naam en paswoord hebt gevraagd is dat nu op die wijze ook makkelijk te controleren of de gebruiker in je pass-file staat. Met een gewone if-constructie in te lassen is dat zo gepiept. Laten we nu eens stellen dat de gebruiker als naam joris en paswoord 1234567 had ingegeven, kunnen we het dan zo controleren :

Voorbeeld met controle : mijnpasscon.php -- broncode mijnpasscon.php

```
<?php
$naam="joris";
$paswoord="1234567";
require("header.php");
$mijn_file = fopen ("mijnpass.txt", "r");
while (!feof ($mijn_file)) {
    $file_inhoud = fgets($mijn_file);
    $file_inhoud = explode(",",$file_inhoud);
    $naamcontrole=trim($file_inhoud[0]);
    $passcontrole=trim($file_inhoud[1]);
    if ($naamcontrole==$naam && $passcontrole==$paswoord) { $goed = "ok"; }
}
fclose ($mijn_file);
if ($goed != "ok") { echo "Fout paswoord !<br>\n";
                                exit; }
?>

<strong>Als het paswoord correct is, gaan we hier gewoon verder.</strong>

<?php
}
?>
<?php
require("footer.php");
```

?gt;

We controleren nu met onze if lus of de naam en het paswoord kloppen, indien ja, dan plaatsen we de variabele $goed op "ok". Dus, zolang er geen namen en paswoorden gevonden worden die corresponderen, blijft de variabele $goed ledig. Nadat de file werd doorlopen en de variabele $goed blijft ledig, dan printen we de message "Fout paswoord !" en we voegen een exit toe zodat het php script verlaten wordt. Indien er wel een goed paswoord en username werd gevonden, gaan we gewoon verder met onze pagina.

Pas het script maar eens aan met een goede usernaam en paswoord uit je pass-file en je zult zien dat je dan gewoon het bericht ziet : **Als het paswoord correct is, gaan we hier gewoon verder.**

Een grafische teller

Wat zou je denken van een mooie grafische teller op je site ? Met PHP is dat heel makkelijk. We hebben in de cursus gezien dat we met include of require, code in onze pagina kunnen invoegen die elders staat. We gaan daar nu gebruik van maken om een funktie te schrijven, die voor ons onze teller toont.

Wat gaan we aanmaken ?

- Een teller file die we pagteller.txt gaan noemen
- Grafische tellertjes
- Een funktie voor onze teller

Onze pagteller.txt file

We maken met onze tekseditor deze file met als inhoud 25791.

Sla deze file op, en voor de unix gebruikers, doe een chmod 666 op die file.

Nu hebben we nog gif bestandjes nodig voor onze getallen. Die gaan we 0.gif tot en met 9.gif noemen. Ik toon hier dewelke ik gebruik :

Nu, ik ben geen grafische expert. Ik heb die buttons gewoon aangemaakt met The Gimp. Dit gaat heel eenvoudig in zijn werk. Iedere unix gebruiker zal The Gimp wel op zijn schijf staan hebben, als je nu Windows gebruikt als besturingssysteem, geen probleem, The Gimp bestaat immers ook voor Windows en is helemaal gratis de downloaden en te gebruiken.

Hoe begin je eraan ? Wel om te beginnen start The Gimp op. Dan kies je het menu Xtns=>Script-Fu=>Web Pages Themes=>Alien Glow=>Button...

Speel nu maar wat met The Gimp totdat je een teller hebt die je aanstaat. Intussen gaan we wat verder coden :-)

Nu onze funktie. Programmeurs hebben er een grote hekel aan om steeds dezelfde code terug te schrijven. We hebben nu een teller, en we willen deze tonen op elke webpagina. Je kunt natuurlijk dan de code op elke pagina herhalen, maar zoals gezegd, we moeten daarvoor een andere oplossing vinden. Om daaraan tegemoet te komen hebben de funkties ter onzer beschikking.

PHP komt al standaard met heel wat funkties die we kunnen gebruiken, en waarvan we in deze cursus al ruim gebruik hebben gemaakt. Funkties zoals fopen(), fread(), count(), date() enz .... zoals je ziet wordt een funktie dus aangeroepen met zijn naam en de argumenten tussen de haakjes. Wel, wij

gaan nu gewoon hetzelfde doen, we gaan ook een funktie schrijven met wat wij willen dat hij doet, de enige vereiste die daarvoor nodig is, je moet PHP vertellen waar die funktie staat en daarvoor hebben we dan include of require voor.

We hebben nu onze pagteller.txt met inhoud 25791. We willen nu dat dit getal getoond wordt op elke pagina. We zullen dus een funktie schrijven daarvoor dat we lesfunktie.php gaan noemen en waarin we de code opnemen. De funktie zelf, gaan we gewoon teller noemen, we kunnen deze dan aanroepen op elke pagina die we wensen met de instructie teller().

Broncode lesfunktie.php

```
<?php function teller($file) { $pointer=fopen("$file","r+"); $teller=fgets ($pointer,11); //we verwijderen de eventuele white spaces $teller=trim ($teller); fclose($pointer); echo "<div align=\"center\"><table cellpadding=\"0\" cellspacing=\"0\"><tr>"; // We bepalen hoe lang ons getal is $lengte=strlen($teller); // We halen nu één voor één een cijfer uit onze tellerfile // Naargelang de waarde van het getal tonen we de imagefile // dat verbonden is aan dat getal for($i=0;$i<$lengte;$i++) { $pos=substr ($teller,$i,1); echo '<td><img src="images/' . $pos . '.gif"></td>'; } echo "<br></tr></table></div>"; } ?>
```

Dit is allemaal bekende code voor ons. Het enige wat hier opvalt is het begin van de file. Daar plaatsen we :

function teller($file) { }

Daarmee geven we te kennen dat onze funktie teller heet en dat hij één argument meekrijgt, ($file). De code van de funktie wordt dan tussen de accolades geschreven.

Nu die $file ? Waar halen we die vandaan ? Wel heel eenvoudig, we sturen de waarde van $file mee vanuit onze pagina waar we de funktie aanroepen. In ons geval gaat $file de waarde pagteller.txt aannemen, zijnde onze teller file. In onze webpagina gaan we dus gewoon de teller aanroepen als volgt : **teller($naam_van_onze_tellerfile);**. Je ziet hier meteen de kracht van dergelijke funktie. Je kunt die teller funktie gebruiken voor om het even welke teller file, het enige wat je moet doen, is vertellen aan de funktie teller welke file je wilt zien.

We gaan dit nu allemaal eens in een webpagina gieten die ik pag1.php noem -- broncode pag1.php

```
<?php require("header.php"); require("lesfunktie.php"); ?> <div align="center"><h2>Dit is onze pag1.php pagina</h2></div> <!-- We gaan nu onze teller funktie oproepen om onze teller te tonen. --> <?php $tellerfile="pagteller.txt"; teller($tellerfile); ?> <p>Nadat onze teller is getoond, gaan we gewoon verder met onze pagina.</p> <p>Je kunt dus om het even waar in je pagina die funktie oproepen.</p> <p>Ik wil die namelijk hier ook nog eens zien. Ik roep gewoon de funktie opnieuw op, en de teller wordt opnieuw getoond.</p> <?php $tellerfile="pagteller.txt"; teller ($tellerfile); ?> <?php require("footer.php"); ?>
```

Nu, onze teller gaat hier gewoon de inhoud van de tekst file pagteller.txt tonen. We willen natuurlijk ook dat de teller met één verhoogd wordt en dat de nieuwe waarde wordt getoond. Wel, gewoon de funktie lesfunktie.php wat aanpassen, zodat dat voor elkaar komt.

Pas nu op ! Ik ga hier een nieuwe funktie aanmaken, namelijk lesfunktie2.php. Ik doe dit hier enkel en alleen om jullie de verschillen te tonen. Jullie blijven gewoon jullie lesfunktie.php gebruiken om tot hetzelfde resultaat te komen. Ik zal echter nu in mijn webpaginas require ("lesfunktie2.php"); moeten opnemen voor ons volgend voorbeeld. Ik zal dit telkens doen als de funktie veranderd. Jullie moeten echter alles in één file houden. Ik zal ook nog een pagteller2.txt aanmaken. Zo ga je de kracht

zien van de funkties.

Broncode lesfunktie2.php

```
<?php function teller ($file) { $pointer=fopen("$file","r+"); $teller=fgets ($pointer,11); //we verwijderen de eventuele white spaces $teller=trim ($teller); ############################################ ## Dit hier toevoegen aan je lesfunktie.php # ############################################ // Nu gaan we onze teller met één verhogen $teller++; // We gaan nu onze nieuwe waarde wegschrijven in onze teller file rewind($pointer); fputs($pointer,$teller); ########################################## ## alles hieronder blijft verder hetzelfde # ########################################## // en we sluiten onze file netjes af fclose($pointer); echo "<div align=\"center\"><table cellpadding=\"0\" cellspacing=\"0\"><tr>"; // We bepalen hoe lang ons getal is $lengte=strlen($teller); // We halen nu één voor één een cijfer uit onze tellerfile // Naargelang de waarde van het getal tonen we de imagefile // dat verbonden is aan dat getal for ($i=0;$i<$lengte;$i++) { $pos=substr($teller,$i,1); echo '<td><img src="../images/' . $pos . '.gif"></td>'; } echo "<br></tr></table></div>"; } ?>
```

Nu maak ik een pag2.php aan waarin ik de gevens van pag1.php overneem, en dan het verschil laat zien.

Voorbeeld pag2.php -- broncode pag2.php

```
<?php require("header.php"); require("lesfunktie.php"); require("lesfunktie2.php"); ?> <div align="center"><h2>Dit is onze pag2.php pagina</h2></div> <!-- We gaan nu onze teller funktie oproepen om onze teller te tonen. --> <?php $tellerfile="pagteller.txt"; teller($tellerfile); ?> <p>Nadat onze teller is getoond, gaan we gewoon verder met onze pagina.</p> <p>Je kunt dus om het even waar in je pagina die funktie oproepen.</p> <p>Ik wil die namelijk hier ook nog eens zien. Ik roep gewoon de funktie opnieuw op, en de teller wordt opnieuw getoond.</p> <?php teller($tellerfile); ?> <p>Nu roep ik mijn tweede funktie aan die ik als voorbeeld heb gegeven.</p> <p>Door het aanroepen van de funktie, zou de paginateller nu steeds met één moeten verhogen, van zodra je een refresh van de pagina doet.</p> <?php $teller2file="pagteller2.txt"; teller2($teller2file); ?> <?php require ("footer.php"); ?>
```

We gaan nu onze final_funktie.php aanmaken. Ik wil echter nog een funktie toevoegen. Namelijk de funktie essetee(). Die gaat dan gewoon wat info printen over mezelf. Dit om te tonen hoe alles werkt. De nieuwe funktie gaat dan essetee heten en de code wordt terug opgenomen tussen de accolades.

Broncode final_funktie.php

```
<?php function teller($file) { $pointer=fopen ("$file","r+"); $teller=fgets($pointer,11); //we verwijderen de eventuele white spaces $teller=trim($teller); ################################### ## Dit hier toevoegen aan je lesfunktie.php # ################################### // Nu gaan we onze teller met één verhogen $teller++; // We gaan nu onze nieuwe waarde wegschrijven in onze teller file rewind($pointer); fputs($pointer,$teller); ################################## ## alles hieronder blijft verder hetzelfde # ################################## // en we sluiten onze file netjes af fclose($pointer); echo "<div align=\"center\"><table cellpadding=\"0\" cellspacing=\"0\"><tr>"; // We bepalen hoe lang ons getal is $lengte=strlen($teller); // We halen nu één voor één een cijfer uit onze tellerfile // Naargelang de waarde van het getal tonen we de imagefile // dat verbonden is aan dat getal for($i=0;$i<$lengte;$i++) { $pos=substr ($teller,$i,1); echo '<td><img src="../images/' . $pos . '.gif"></td>'; } echo "<br></tr></table></div>"; } ###################################### ## Ik voeg hier gewoon een nieuwe funktie toe # ###################################### function essetee() { echo
```

"

De auteur van deze cursus is : Serge Terryn
";
echo "URL : <a
href=\"http://www.essetee.be\">http://www.essetee.be
";
echo "MSN : <a
href=\"mailto:essetee@hotmail.com\">essetee@hotmail.com
";
echo "ICQ : 763290
"; } ?>

En we maken nu een nieuwe webpagina aan pag3.php -- broncode pag3.php

<?php require("header.php"); require("final_funktie.php"); ?> <div
align="center"><h2>Dit is onze pag3.php pagina</h2> <!-- We gaan nu
onze teller funktie oproepen om onze teller te tonen. --> De waarde van
je teller zou hier nu moeten staan. Telkens je reload doet moet hij nu met
eentje verhogen :-) <?php $tellerfile="pagteller2.txt"; teller
($tellerfile); ?> En we roepen een bijkomende funktie op, namelijk
essetee() . <?php essetee(); ?> En die funktie toont dus gewoon wat
info over mij. <?php require("footer.php"); ?>

Op die wijze kan ik overal in mijn paginas die info plaatsen door gewoon de funktie essetee() aan te roepen. Onthoud ! Steeds het juiste pad vermelden naar de plaats waar je funktie staat, zoniet ga je errors krijgen voor het aanroepen van een onbekende funktie. Gebruik daarom steeds require om je funktie te includen. Als het pad verkeerd staat, gaat require dat melden. Mocht je include gebruiken, ga je enkel een fout krijgen waar je niet veel wijzer uit wordt als je pas begint met PHP.

Tot zover onze teller en het schrijven van een funktie.