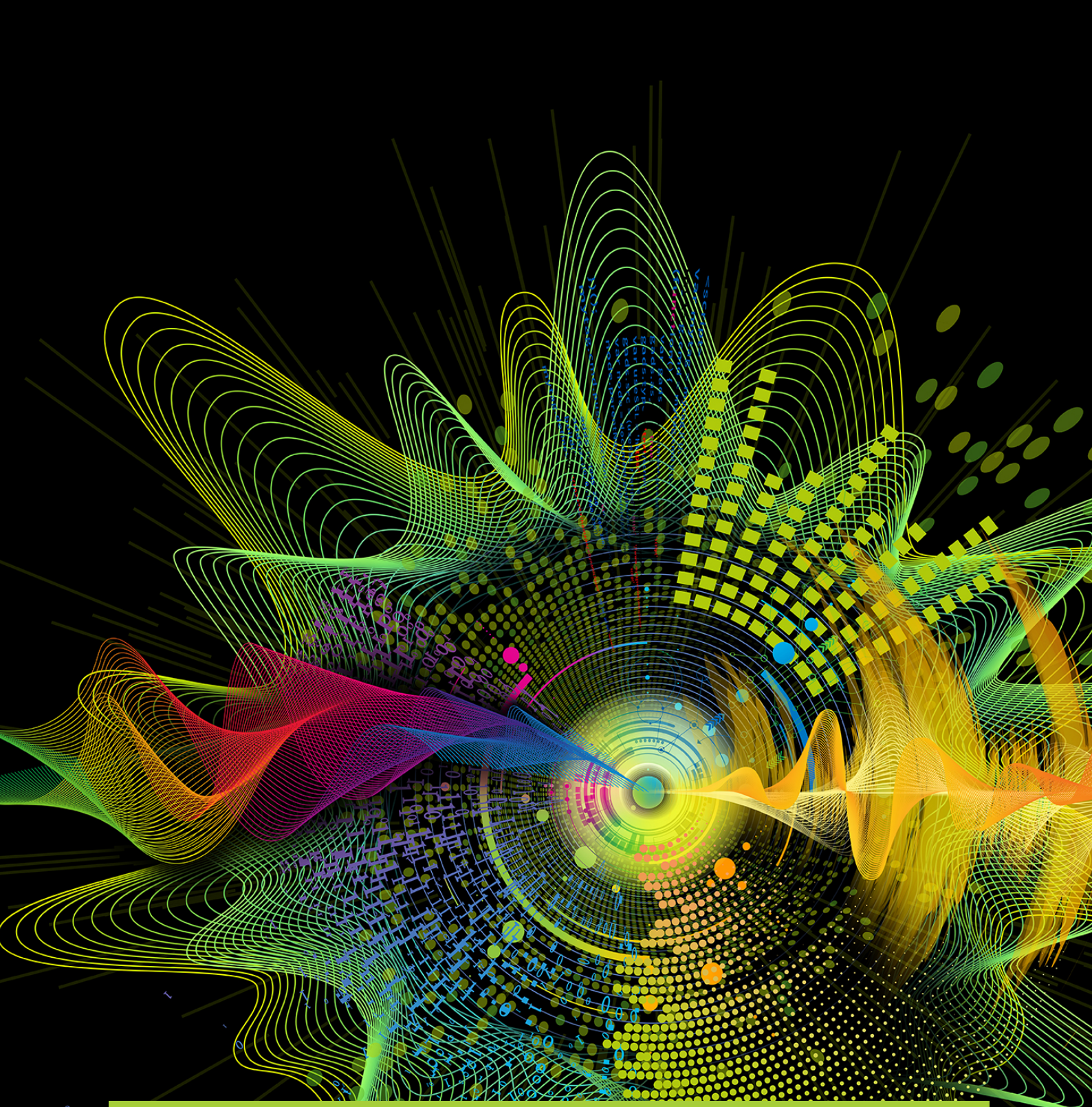


Verdieping programmeren

Module 1: PHP 1

Leerjaar:	1
Crebonummer:	25604
Sector:	Software-developer

INTRODUCTIE

In de vorige periode hebben jullie kennis gemaakt met HTML, CSS en JavaScript. Hiermee kan je eenvoudige statische pagina's maken. 'Statisch' betekent dat elke keer wanneer je een bepaalde url opvraagt, je dezelfde htmlpagina terugkrijgt.

Een website wordt 'dynamisch' als de content van de pagina te veranderen is; hetzij door de gebruiker, hetzij door de code zelf. 'Dynamisch' betekent dat de pagina die je terugkrijgt op de server op dat moment wordt gegenereerd en dus elke keer anders kan zijn. Neem als voorbeeld een pagina die een lijst met gebruikers weergeeft. Hoe meer gebruikers zich hebben aangemeld, hoe meer gebruikers er op de pagina staan die je terugkrijgt van de server. Een (script-)taal waarmee je zo'n dynamische website kan bouwen heet **PHP**. Bij het ontstaan van PHP stond de afkorting voor 'Personal Home Page' maar dit is veranderd naar 'PHP: Hypertext Preprocessor'.

In deze module leer je de basis van PHP**.** Je leert o.a. werken met variabelen, hoe je een formulier kunt uitlezen en hoe je de code goed kunt structureren.

Aan het eind van deze module kan je een eenvoudige dynamische website maken waarbij interactie plaatsvindt tussen de gebruiker en de website.

De leerdoelen voor deze module zijn:

Je kunt:

a) een PHP-pagina maken, opslaan en uploaden naar een webserver.
b) een PHP-pagina maken met de juiste syntax.
c) gegevens op het scherm tonen met het echo-commando.
d) in PHP variabelen declareren, initialiseren en gebruiken.
e) beslisstructuren toepassen in jouw PHP-code.
f) in PHP herhalingslussen maken.
g) invoervelden uitlezen met behulp van PHP.

TIJDSPLANNING

Week 1		
MINOR1, les 1	2 uur	Hoe werkt PHP, Oefening 1+2, Opdracht 1
Week 2		
MINOR1, les 2	2 uur	Variabelen, arrays, Oefening 3+4+5, Opdracht 2
Week 3		
MINOR1, les 3	2 uur	Interactie met de gebruiker, Oefening 6+7, Eindopdracht
		Verdiepingsopdracht

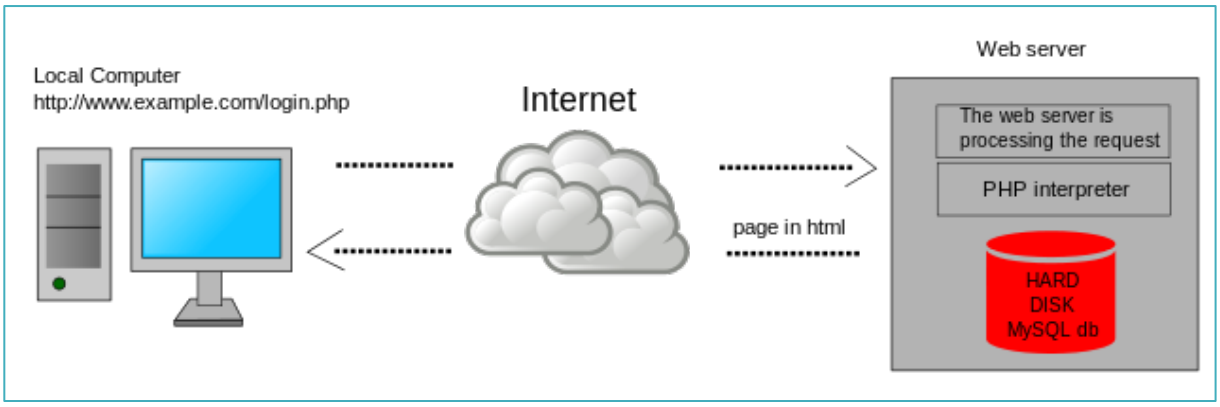
LES 1 - 2 uur

HOE WERKT PHP

INLEIDING

In de vorige perioden hebben jullie geleerd hoe je een HTML-pagina moet maken en hoe je JavaScript kunt gebruiken om interactie met de HTML-pagina te verzorgen. Als je een pagina hebt gemaakt dan sla je hem op als (bijvoorbeeld) pagina1.html. Door erop te dubbelklikken wordt de pagina geopend in een browser. Uiteindelijk moet de gemaakte pagina op een webserver geplaatst worden zodat je werk nagekeken kan worden door de docenten maar voor de werking van de pagina maakt het niet uit waar je de HTML-pagina opslaat. Deze manier van werken wordt *cliënt-side of front-end development* genoemd.

Voor een PHP-pagina werkt dat anders: de pagina MOET worden opgeslagen op een webserver, een computer die weet hoe hij met PHP moet omgaan. Deze server interpreteert de PHP-code via speciale PHP-software (een *interpreter*), bewerkt de pagina en stuurt het geheel terug als HTML-code. Deze HTML-code kan weer gelezen worden door de browser.



1. Vanaf de lokale computer (de cliënt) wordt via een browser een PHP-pagina opgevraagd.
2. De webserver ontvangt het verzoek om de PHP-pagina te mogen bekijken.
3. De PHP-software op de webserver interpreteert de code, indien nodig worden er gegevens uit de database toegevoegd.
4. De webserver stuurt het resultaat in HTML-code terug naar de cliënt.
5. De browser van de cliënt laat de, in HTML-code omgezette, pagina zien.

De cliënt kan dus nooit de PHP-code zien, maar alleen gegenereerde HTML-code. Deze manier van werken wordt *server-side of back-end development* genoemd.

Ter info: Een developer die zowel server- als cliënt-side development doet wordt ook wel een *full-stack developer* genoemd.

EXTERNE BRONNEN

Bekijk dit filmpje voor een korte uitleg over wat PHP is en wat je nodig hebt om met PHP aan de slag te gaan https://www.youtube.com/watch?v=kY5P9sZqFas

In dit filmpje op LinkedIn Learning wordt uitgelegd hoe PHP werkt https://www.linkedin.com/learning/learning-php-4/how-php-works?u=36995692

Lees dit artikel voor extra informatie over PHP https://html-site.nl/wat-is-php/

Extra uitleg over Client-Side vs Server-Side Web Development vind je op https://careerkarma.com/blog/client-vs-server-side-development/

WAT HEB JE NODIG

Om met PHP-scripting te beginnen heb je een aantal zaken nodig:

- Webserver met PHP-ondersteuning

  Op jouw webserver op school is PHP geïnstalleerd. Je hoeft dus zelf niets extra's te doen om PHP-bestanden te kunnen openen. Zoals je weet is jouw website in de browser bereikbaar via *http://<stnr>.ict-lab.nl* waarbij je *<stnr>* vervangt voor jouw *eigen studentnummer*.

  Het adres van de webserver op school is *https://web02.ict-lab.nl*

- Een FTP-programma

  Om jouw bestanden op de webserver te plaatsen (uploaden) heb je een FTP-programma nodig. Op school gebruiken we vaak FileZilla maar er zijn natuurlijk nog veel meer programma's die je hiervoor kunt gebruiken.

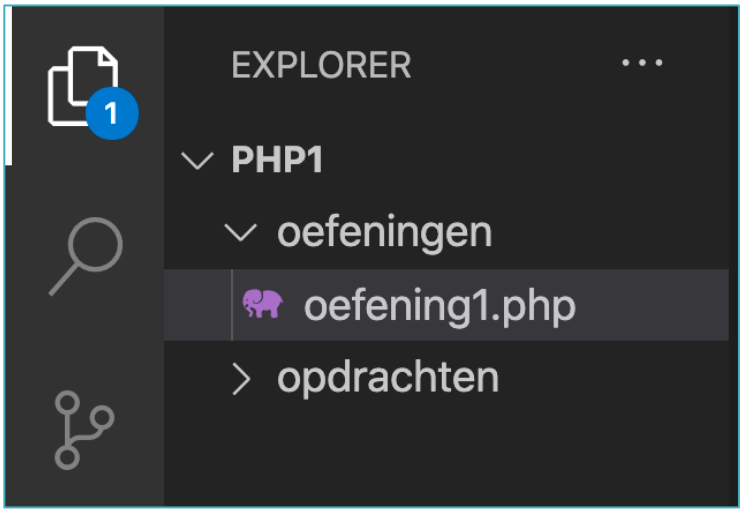
- Een editor

  Het maken (schrijven) van een PHP-script doe je in een editor. Tijdens de opleiding maken we vaak gebruik van Visual Studio Code maar ook hiervoor geldt dat er nog veel meer andere programma's zijn.

Via de site CodePHPOnline kan je heel makkelijk PHP-code uitproberen zonder dat je een webserver nodig hebt. Ideaal om te oefenen! https://www.codephponline.com/

BELANGRIJKE AFSPRAKEN

- De oefeningen en opdrachten voor deze module sla je op als een PHP-bestand. De pagina's die je maakt hebben allemaal een *.php extensie*.
- De oefeningen en opdrachten voor deze module sla je op in een map met een logische naam. Noem de map *PHP1* en maak daarin een aparte map voor de *oefeningen* en een aparte map voor de *opdrachten*.



- Je maakt jouw PHP-pagina lokaal (*op de cliënt*). Voordat je de PHP-pagina kan testen moet je deze op de server zetten (*uploaden*). Daarna kan je de juiste URL intypen in de browser. Bijvoorbeeld *http://12345.ict-lab.nl/php1/test.php*.

  Je werkt dus lokaal, maar je test op de server. Door op deze manier te werken heb je ook altijd een backup van je bestanden op jouw eigen computer of SSD.

OEFENING 1

Je weet nu globaal hoe PHP werkt en wat je nodig hebt om PHP-scripts te kunnen schrijven. Neem onderstaande code over en sla de pagina op als *oefening1.php*.

oefening1.php ●

oefeningen > oefening1.php

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <title>Mijn eerste PHP-pagina</title>
7   </head>
8   <body>
9       <?php
10          echo "Welkom op mijn eerste PHP-pagina";
11      ?>
12  </body>
13  </html>
14
```

Upload de pagina naar jouw webserver naar de map *php1/oefeningen*. Open het bestand in een browser door de volgende URL in te typen: *http://<stnr>.ict-lab.nl/php1/oefening1.php*

Natuurlijk vervang je *<stnr>* door jouw eigen *studentnummer*.

Als je de code goed hebt overgenomen dan zie je de volgende uitvoer op je scherm:

minderhoud.ict-lab.nl/basis1b_1c/P3/php1/oefeningen/oefening1.php

Welkom op mijn eerste PHP-pagina

Laten we eens kijken naar de code van *oefening1.php*:

- De structuur van de pagina is gemaakt met HTML.
- De PHP-code staat tussen de HTML-tags *<?php* en *?>*. Deze tags vertellen de PHP-software op de server dat er PHP-code staat tussen de openingstag *<?php* en de sluittag *?>*.
- Het commando *echo* zorgt voor de uitvoer naar de pagina.
- De letterlijke tekst die getoond wordt staat tussen dubbele aanhalingstekens (*double-quotes*).
- Elke opdrachtregel in PHP wordt afgesloten met een *puntkomma*.

OEFENING 2

Zoals je in oefening 1 hebt gezien mag je PHP-code tussen HTML-code schrijven zolang de PHP-code maar tussen PHP-tags staat. Je kunt bij de uitvoer naar het scherm (echo) ook HTML-tags opnemen in het echo-statement. In deze oefening gaan we daar mee aan de slag. Neem onderstaande code over, sla de pagina op als *oefening2.php* en upload de pagina naar jouw webserver.

oefeningen > oefening2.php

```
1   <style>
2       #rodeTekst {
3           color:red;
4       }
5
6       #nadruk {
7           text-decoration: underline;
8       }
9
10      #dikGedrukt {
11          font-weight: bold;
12      }
13  </style>
14  <?php
15      echo "<h1>Welkom bij oefening 2</h1>";
16
17      //Dit is de manier om een comment van 1 regel in je PHP-code te zetten
18      echo "Deze pagina heeft <span id=nadruk>geen HTML-structuur</span>. ";
19      echo "De webserver kan de pagina gewoon verwerken.<br>";
20
21      /*Voor een comment die bestaat uit meerdere regels
22      gebruik je deze methode*/
23      echo "In deze oefening gebruik ik <span id=dikGedrukt>meerdere HTML-tags.</span>";
24      echo "<p id=rodeTekst>Deze tekst heeft een kleur</p>";
25
26  ?>
```

Het resultaat van oefening 2:

Welkom bij oefening 2

Deze pagina heeft geen HTML-structuur. De webserver kan de pagina gewoon verwerken. In deze oefening gebruik ik **meerdere HTML-tags.**

Deze tekst heeft een kleur

Onderstaande afbeelding laat de HTML-code zien die door de webserver is teruggestuurd naar de browser.

```
Elements  Console  Sources  Network  Performance  »
<html>
  ▶<head>…</head>
… ▼<body> == $0
    <h1>Welkom bij oefening 2</h1>
    "Deze pagina heeft "
    <span id="nadruk">geen HTML-structuur</span>
    ". De webserver kan de pagina gewoon verwerken."
    <br>
    "In deze oefening gebruik ik "
    <span id="dikGedrukt">meerdere HTML-tags.</span>
    <p id="rodeTekst">Deze tekst heeft een kleur</p>
  </body>
</html>
```

Laten we ook deze code eens bekijken:

- De pagina heeft geen duidelijke HTML-structuur. We zien geen <*head*> en <*body*>, zelfs de <*html*> tag ontbreekt. Dit kan en mag, voor de werking van PHP is het niet noodzakelijk om een nette HTML-structuur te hebben in je pagina. Als we via de Inspector kijken naar de code die teruggestuurd is door de webserver zien we echter wel een HTML-structuur, dit heeft de browser voor ons gedaan.
- We gebruiken in de verschillende echo commando's een aantal HTML-tags zoals een <*h1*> op regel 15. De PHP-software op de webserver herkent deze tags en zal ze invoegen in de HTML-code die teruggestuurd wordt naar de browser.
- Op regel 18 zien we het HTML-element <*span*>, dit element heeft een id gekregen. In de CSS-code boven in de pagina gebruiken we dit id om opmaak te geven aan het <*span*> element. Op deze manier kunnen we de opmaak van de pagina doen zoals we dat gewend zijn namelijk met CSS.
- Verder zien we ook 2 manieren om commentaarregels te gebruiken in PHP-code namelijk:
    - '//' De 'double-slash' – hiermee kan je een 1 commentaarregel in je code zetten.
    - '/* */' De 'slash-star' – hiermee kan je commentaar plaatsen over meerdere regels heen.

PHP IN HTML

In de oefeningen hierboven hebben we het echo-commando gebruikt om tekst op het scherm te laten schrijven inclusief HTML-tags. Het nadeel van deze manier is dat je vanuit de editor geen code-hints meer krijgt bij het schrijven van HTML. We laten jullie wel werken met deze manier van coderen omdat het helpt bij het begrijpen van de mogelijkheden en de werking van het echo-commando.

Er is, natuurlijk, ook een andere manier om HTML en PHP te mixen en dat is door de PHP-code tussen je HTML te schrijven. Huh wat? Kijk maar eens naar het volgende voorbeeld:

```
1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <meta charset="UTF-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1.0">
6          <title>Document</title>
7          <style>
8              #nadruk {
9                  text-decoration: underline;
10             }
11         </style>
12     </head>
13     <body>
14         <h1>Voorbeeld PHP in HTML</h1>
15         <p>De basis van deze pagina is <span id=nadruk>HTML</span>.</p>
16         <p>Als ik PHP-code wil gebruiken binnen HTML dan moet ik die code tussen de PHP-tags opnemen</p>
17         <p>Vandaag is het <?php echo date('d M Y');?></p>
18
19         <!-- Je kunt PHP en HTML mixen -->
20         <ul>
21             <?php for($i=1;$i<=5;$i++){ ?>
22                 <li>Onderdeel <?php echo $i; ?></li>
23                 <?php } ?>
24         </ul>
25     </body>
26 </html>
```

In regel 17 zie je een stukje PHP-code dat midden in een regel HTML-code staat. De PHP-code staat tussen de PHP-tags waardoor de code netjes verwerkt zal worden door de server. Op de pagina zal de datum van vandaag te zien zijn.

In regel 21 maken we met PHP een *for-lus* om een lijstje met 5 onderdelen te tonen. Zoals je ziet openen en sluiten we een aantal keer de PHP-tags zodat we HTML en PHP kunnen mixen.

Check onderstaande bronnen voor extra uitleg over het schrijven van PHP in HTML:

https://www.ntchosting.com/encyclopedia/scripting-and-programming/php/php-in/

https://www.youtube.com/watch?v=T7dIIVodwec&feature=youtu.be

ZELF AAN DE SLAG

Na deze eerste kennismaking met PHP ga je nu zelfstandig verder leren en oefenen met PHP. Hiervoor ga je gebruik maken van de volgende twee *tutorials*:

- W3School PHP Tutorial - https://www.w3schools.com/php/default.asp
    - Start met 'PHP home' en werk door tot en met 'PHP Arrays' zodat je de kennis hebt om opdracht 1 en 2 te maken.
- LinkedIn Learning - https://www.linkedin.com/learning/learning-php-4/
    - Start met '1. The Basics' en werk door tot en met '4. Loops' zodat je de kennis hebt om opdracht 1 en 2 te maken.

OPDRACHT - LES 1

OPDRACHT 1, INDEX.PHP

Maak een PHP-pagina die je opslaat als *index.php*. Op deze pagina worden de volgende gegevens getoond:

- Jouw naam;
- Je studentnummer;
- Je klas
- De naam van 4 leerlijnen van jouw opleiding met daarachter het cijfer dat je op dit moment hebt voor die leerlijn.

Eisen

- Alle teksten worden op het scherm gezet met *PHP-code*.
- Alle vormgeving (kleuren, dikke gedrukte tekst, etc) wordt met *CSS* gedaan.
- Waar mogelijk maak je gebruik van HTML-tags (b.v. *<h3>*, *
*, *<p>).*

Leerdoelen

Je kunt:

a) een PHP-pagina maken, opslaan en uploaden naar een webserver.
b) een PHP-pagina maken met de juiste syntax.
c) gegevens op het scherm tonen met het echo-commando.

LES 2 - 2 UUR

VARIABELEN

THEORIE

Net als in JavaScript kun je in PHP ook gebruik maken van variabelen om gegevens op te slaan zodat je deze gegevens later weer kunt gebruiken in je code. Door variabelen te gebruiken kan je jouw website laten reageren op bijvoorbeeld invoer van de bezoeker van jouw website. Een groot voordeel van PHP ten opzichte van JavaScript is dat PHP de ingevoerde gegevens ook kan wegschrijven naar een database, zodat de gegevens ook later weer te gebruiken zijn.

Voor het aanmaken en gebruiken van variabelen in PHP gelden de volgende regels:

- Een variabele begint met het *$-teken* (dollar-sign), gevolgd door de naam van de variabele.
- De naam van een variabele moet beginnen met een *letter* of een *underscore* (laag streepje).
- De naam van een variabele mag *niet* met een *cijfer* beginnen.
- In de naam van een variabele mogen alleen *cijfers*, *letters* en een *underscore* voorkomen.
- De naam van een variabele is *hoofd- en kleineletter* gevoelig (case-sensitive). *$Naam* is dus een andere variabele dan *$naam*.

Het toekennen van een waarde aan een variabele gebeurt met het = *teken*.

```
<?php
$tekst = "Welkom bij PHP1";     //Plaats de string in de variabele $tekst
$getal1 = 10;                   //Ken de waarde 10 toe aan de variabele $getal1
$getalTwee = 10.5;              //Geeft $getalTwee de waarde 10.5

?>
```

PHP kent verschillende typen variabelen ook wel data types genoemd. De meeste zal je herkennen vanuit JavaScript. PHP is een *weak-typed* scripting-taal. Dit houdt in dat je bij het aanmaken van een variabele in PHP niet hoeft aan te geven van welke type de variabele is. Dit bepaalt PHP zelf op basis van de waarde die je aan de variabele toekent.

Extra uitleg over de verschillende data types vind je in dit artikel
https://www.w3schools.com/php/php_datatypes.asp

OEFENING 3

Om de inhoud van een variabele te tonen maken we gebruik van het echo-commando. Neem onderstaande code over in een nieuwe PHP-pagina en sla deze op als *oefening3.php*. Vergeet niet om de pagina te uploaden naar jouw webserver.

```
oefeningen > oefening3.php
1   <?php
2   $tekst = "Welkom bij PHP1";      //Plaats de string in de variabele $tekst
3   $getal1 = 10;                    //Ken de waarde 10 toe aan de variabele $getal1
4   $getalTwee = 10.5;               //Geeft $getalTwee de waarde 10.5
5
6   echo $tekst;
7   echo "<br>";
8   echo $getal1;
9   echo "<br>";
10  echo "Dit is een float: " . $getalTwee;
11
12  ?>
```

In regel 6 gebruiken we het echo-commando om de inhoud van de variabele *$tekst1* op het scherm te laten zien. De inhoud van deze variabele is een string, dat hebben we gedaan door in regel 2 de tekst tussen *double-quotes* te zetten.

Wat je misschien opvalt is dat we in regel 10 eerst een tekst op het scherm zetten waar we met een punt (.) de variabele $getalTwee aan vast plakken. Dit noemen we *concateneren*. In JavaScript gebruiken we het plus-teken (+) om te concateneren.

ARRAYS

THEORIE

In de module JavaScript 2 heb je geleerd dat een *array* niets anders is dan een *lijst met gegevens*. Je kunt het zien als een variabele die meerdere waarden bevat. In PHP wordt veel gebruik gemaakt van arrays. Met name bij het uitlezen van gegevens uit een database maar ook voor andere handelingen is het heel handig om gebruik te maken van arrays.

Een array bestaat uit één of meerdere *entrie/elementen* die allemaal een eigen *sleutel/index* en een *waarde/value* hebben. Als je bij het declareren van een array geen sleutel meegeeft aan de waarden, genereert PHP er zelf een. De eerste sleutel is dan 0, de tweede 1, enzovoort. Dit type array noemen we een *Indexed Array*.

Het declareren van een array gaat bijna hetzelfde als het declareren van een variabele. Alleen geven we nu de variabele niet een enkele waarde zoals een string of integer, maar kennen we een lijst van waarden toe.

OEFENING 4

In de deze oefening gaan we aan de slag met arrays. Neem onderstaande code over in een nieuwe PHP-pagina en sla deze op als *oefening4.php*. Vergeet niet om de pagina te uploaden naar jouw webserver.

```
oefening3.php ×
oefeningen > oefening3.php
1  <?php
2  //Maak de array en vul hem met waarden
3  $mijnGames = array('Mario Kart', 'Fortnite', 'Apex Legends', 'Patience');
4
5  //Print de volledige array naar het scherm
6  print_r($mijnGames);
7
8  //Uitlezen van een waarden uit een array
9  echo "<p>Mijn favoriete game is " . $mijnGames[3] . ".</p>";
10 echo "<p>Geintje, het is " . $mijnGames[0] . ".</p>";
11
12 //Een volledige array uitlezen
13 foreach ($mijnGames as $game) {
14     echo $game . " is een geweldig spel! <br>";
15 }
16 ?>
```

Probeer voor jezelf te bedenken wat de verschillende code-regels doen. Als dit nog onduidelijk is kijk dan in de tutorial die je aan het volgen bent naar het onderwerp *Arrays*.

Nu we weten hoe we een array kunnen maken en uitlezen kijken we nog even naar het toevoegen van elementen aan een array.

Om een enkele waarde toe te voegen roepen we de naam van de array aan met direct daar aan vast *lege blokhaken*.

```
//1 element toevoegen aan een array
$mijnAutos[] = 'Alfa Romeo';
```

Soms wil je op een snelle manier meerdere waarden toevoegen aan een array. Je kunt dan bovenstaande code een aantal keer kopiëren en plakken maar er is een handigere manier om dit te doen.

```
//Meerdere waarden toevoegen aan een array
array_push($mijnAutos, 'Maserati', 'Fiat', 'Morgan');
```

OEFENING 4 UITBREIDEN

Open *oefening4.php* en breidt de array *$mijnGames* uit met minimaal 3 games. Maak gebruik van allebei de manieren die hierboven zijn uitgelegd. Zorg dat de inhoud van de array weer getoond wordt. Bedenk zelf hoe je de inhoud van de array op jouw pagina kunt laten zien.

ZELF AAN DE SLAG

Nu ga je weer zelfstandig verder leren en oefenen met PHP met de tutorial waar je vorige week mee bent gestart.

- W3School PHP Tutorial - https://www.w3schools.com/php/default.asp
    - Start met 'PHP Home' en werk door tot en met 'PHP Arrays' zodat je de kennis hebt om opdracht 1 en 2 te maken.
- LinkedIn Learning - https://www.linkedin.com/learning/learning-php-4/
    - Start met '1. The Basics' en werk door tot en met '4. Loops' zodat je de kennis hebt om opdracht 1 en 2 te maken.

OPDRACHT - LES 2

OPDRACHT 2, INDEX.PHP UITBREIDEN

Open jouw *index.php*. Je gaat de pagina aanpassen en uitbreiden met het volgende:

- Jouw naam, studentnummer, klas en de 4 leerlijnnamen en leerlijncijfers zijn *variabelen*. Deze gegevens worden overzichtelijk op het scherm getoond.
- De 4 cijfers staan ook in een *array* met de naam *$mijnCijfers*.
- Er komt een extra uitvoer van alle cijfers op een rij, bijvoorbeeld:

Mijn volledige cijferlijst: 6.5 - 7 - 7.2 - 8

Eisen

- Jouw naam, studentnummer, klas en de 4 leerlijnnamen en leerlijncijfers zijn variabelen.
- De inhoud van de variabelen wordt op het scherm gezet met PHP-code.
- Gebruik een *loop* om de array $mijnCijfers uit te lezen en de lijst met cijfers op het scherm te tonen.
- Alle vormgeving (kleuren, dikke gedrukte tekst, etc) wordt met *CSS* gedaan.
- Waar mogelijk maak je gebruik van HTML-tags (b.v. *<h3>*, *
*, *<p>).*

Leerdoelen

Je kunt:

a) een PHP-pagina maken, opslaan en uploaden naar een webserver.
b) een PHP-pagina maken met de juiste syntax.
c) gegevens op het scherm tonen met het echo-commando.
d) in PHP variabelen declareren, initialiseren en gebruiken.
f) in PHP herhalingslussen maken.

LES 3 - 2 UUR

INTERACTIE MET DE GEBRUIKER

THEORIE

Interactie met de gebruiker van een website kent drie stappen:

- Invoer door de gebruiker;
- Verwerking van de ingevoerde gegevens;
- Uitvoer van het resultaat.

De belangrijkst manier van uitvoeren/tonen van een resultaat met PHP kennen we al, dat is het echo-commando. In deze les gaan we ons focussen op de invoer en het verwerken van gegevens via PHP.

Een PHP-script kan op twee manieren gebruikersinvoer versturen en ontvangen:

- Via de URL, ook wel de *GET-methode* genoemd.
- Via een formulier, ook wel de *POST-methode* genoemd.

DE GET-METHODE

Om gegevens via de URL uit te wisselen maken we gebruik van een *superglobal* variabele genaamd: *$_GET*. Superglobals zijn vooraf gedefinieerde variabelen. Deze variabelen zijn arrays die afhankelijk van de situatie allerlei waarden kunnen bevatten. Deze superglobals zijn overal en in elk script beschikbaar.

URL'S

- Een URL *zonder* data
  - http://www.test.com/index.php
- Een URL *met* data
  - http://www.test.com/index.php?naam=Kees
- *Meerdere* parameters in de URL:
  - http://www.test.com/index.php?naam=Kees&leeftijd=18&stad=Delft

Om data aan een URL toe te voegen typen we na de paginanaam een *vraagteken(?)* gevolgd door de *naam* van de variabele (naam) en de *waarde* die we aan die variabele toekennen (Kees). Dit noemen we ook een *URL-parameter*.

We kunnen ook *meerdere* parameters aan de URL toevoegen. Dit doen we door het *ampersand-teken (&)* te plaatsen tussen de verschillende parameters.

OEFENING 5

Neem onderstaande code over in een nieuwe pagina die je opslaat als *oefening5.html*.

```
oefeningen > oefening5.html > p
1   <h2>Klik op 1 van de links</h2>
2
3   <p>
4       <a href="oefening5_verwerk.php?nummer=1">Link nummer 1</a> <br>
5       <a href="oefening5_verwerk.php?nummer=2">Link nummer 2</a> <br>
6       <a href="oefening5_verwerk.php?nummer=3">Link nummer 3</a> <br>
7   </p>
```

In regels 4 tot en met 6 hebben we een link gemaakt naar de pagina *oefening5_verwerk.php*. Aan deze link hebben we een URL-parameter toegevoegd met de naam *nummer*.

Maak nu nog een nieuwe pagina, noem die *oefening5_verwerk.php* en typ de volgende code in de pagina.

```
oefeningen > oefening5_verwerk.php
1   <?php
2       if(isset($_GET['nummer']))
3       {
4           echo "Je hebt op link nummer " . $_GET['nummer'] . " geklikt.";
5       } else
6       {
7           echo "Je hebt niet op een link geklikt, wat doe je op deze pagina?";
8       }
9   ?>
```

Laten we eens kijken naar de code die we hierboven hebben gemaakt. In regel 2 controleren we via de PHP-functie *isset* of er in de superglobal *$_GET* een element staat met de naam *nummer*. Als dit het geval is dan tonen we de waarde die aan *nummer* is meegegeven via de URL. Kan je voor jezelf bedenken waarom we ook een *else* hebben gemaakt?

Het is belangrijk om te onthouden dat het versturen van gegevens via de GET-methode onveilig is want de gegevens zijn zichtbaar via de URL. Gebruik deze methode dus *nooit* om wachtwoorden of andere gevoelige informatie te versturen.

OEFENING 5 UITBREIDEN

Je gaat *oefening5.php* uitbreiden met een extra URL-parameter. Zorg dat er bij iedere link een extra parameter meegegeven wordt naar de verwerkpagina. Deze parameter krijgt als naam *naam* en als waarde bedenk je zelf 3 verschillende namen. Per link worden er dus *2 URL-parameters* meegestuurd.

Op de verwerkpagina wordt de naam ook uitgelezen en op het scherm getoond.

Hallo, Flip.
Je hebt op link nummer 1 geklikt.

Op deze pagina vind je meer informatie over het versturen van meerdere URL-parameters https://scriptverse.academy/tutorials/php-passing-data-via-url.html

Of Google zelf eens op *Passing variables in a URL PHP*

DE POST-METHODE

Om gegevens via een formulier uit te wisselen maken we gebruik van de SUPERGLOBAL variabele genaamd: $_POST. Deze methode maakt het mogelijk om gegevens op een veilige manier te verzenden naar een verwerk-pagina. Dit is veruit de meest gebruikte manier om data uit te wisselen in PHP.

Een formulier, dat je gewoon kunt maken met HTML, heeft minimaal de volgende attributen:

- Een "action"
  - De naam van het script waar het formulier naar opgestuurd wordt.
- Een "method"
  - De manier van verzenden, bijna altijd "post"
- Eén of meerdere velden ("inputs") met een unieke "name".
- Een submitknop.

OEFENING 6

We gaan een oefening maken met de *POST-methode*. Maak een HTML-pagina, sla die op als *oefening6.html* en kopieer en plak de volgende code in de pagina. In de code maken we een HTML-formulier dat de ingevoerde gegevens doorstuurt naar de pagina *oefening6_verwerk.php* als er op de *submit-knop* wordt geklikt.

```
<!DOCTYPE html>
<html>
<head>
    <title>Oefening 6 formulier</title>
    <meta charset="UTF-8" />
</head>
<body>
    <form action="oefening6_verwerk.php" method="post">
    <!-- De velden van het formulier -->
    <p>
        Naam: <input type="text" name="naam">
    </p>
    <p>
        Leeftijd: <input type="number" name="leeftijd">
    </p>
    <input type="submit" name="submit" value="Versturen">
    </form>
</body>
</html>
```

Maak nu een nieuwe pagina *oefening6_verwerk.php* en neem daarin de volgende code over. Deze code zal je moeten overtypen.

```
oefeningen > oefening6_verwerk.php
1   <?php
2   if (isset($_POST['submit']))
3   {
4       $naam = $_POST['naam'];
5       $leeftijd = $_POST['leeftijd'];
6
7       echo "Hallo "  . $naam . ", jij bent " . $leeftijd . " jaar oud.<br>";
8       echo "Over 10 jaar ben jij " . ($leeftijd+10) . " jaar oud.<br>";
9   } else
10  {
11      echo "Je hebt het formulier niet gevuld!";
12  }
13
14  ?>
```

In regel 2 gebruiken we weer de functie *isset*, dit keer om te controleren of er in de array *$_POST* een element staat met de naam *submit*. We controleren hier of mensen wel op de *Versturen-knop* hebben geklikt. Als dit NIET het geval is dan tonen we een melding op het scherm namelijk: *Je hebt het formulier niet gevuld!*.

Als er WEL op de knop is geklikt dan lezen we de ingevulde velden uit door 2 nieuwe variabelen te maken die we vullen vanuit *$_POST*, dit doen we op regels 4 en 5. We doen dit zodat we de ingevulde naam en leeftijd makkelijk kunnen gebruiken. Op regel 7 en 8 tonen we vervolgens de ingevoerde gegevens.

ZELF AAN DE SLAG

Nu ga je weer zelfstandig verder leren en oefenen met PHP en het werken met Forms.

- W3School PHP FORMS - https://www.w3schools.com/php/php_forms.asp
- LinkedIn Learning - https://www.linkedin.com/learning/learning-php-4/
    - Ga naar 8. Processing Forms

OPDRACHT - LES 3

OPDRACHT 3, HET EINDPRODUCT

Maak een *index.html*. Op deze pagina komt een *formulier* waarop de gebruiker van jouw website de volgende gegevens kan invullen:

- Naam;
- Studentnummer;
- Klas
- De naam van 4 leerlijnen en het cijfer dat de gebruiker op dit moment heeft voor die leerlijn.
- Als de gebruiker op de submit-knop drukt dan wordt de inhoud van het formulier doorgestuurd naar de pagina *index_verwerk.php*

Open jouw *index.php* en sla de pagina op als *index_verwerk.php* of maak een nieuwe pagina als verwerk-pagina. Je gaat de pagina aanpassen en uitbreiden met het volgende:

- Alle variabelen worden gevuld met de waarden die ingevuld zijn in het *formulier*.
    - Tip: *$_POST*
- De leerlijnen en de cijfers worden opgeslagen in 1 of 2 *arrays*.
    - Tip: Je kunt hiervoor gebruik maken van *2 Indexed Arrays* of *1 Associative Array*. Als je de uitdaging aan durft dan kan je gebruik maken van een *Multidimensional Array*.
- Op de pagina wordt het *gemiddelde cijfer* van de ingevulde leerlijnen getoond. Als het cijfer *groter dan of gelijk aan* een 5.5 is moet het *groengekleurd* worden. Als het cijfer *kleiner dan* een 5.5 is moet het *roodgekleurd* zijn.

Eisen

- Je gebruikt 2 pagina's: 1 pagina met het formulier en 1 verwerk-pagina.
- De inhoud van de variabelen wordt op het scherm gezet met PHP-code.
- Alle vormgeving (kleuren, dikke gedrukte tekst, etc) wordt met CSS gedaan.
- Waar mogelijk maak je gebruik van HTML-tags (b.v. <h3>, <br>, <p>).

Leerdoelen

Je kunt:

a) een PHP-pagina maken, opslaan en uploaden naar een webserver.
b) een PHP-pagina maken met de juiste syntax.
c) gegevens op het scherm tonen met het echo-commando.
d) in PHP variabelen declareren, initialiseren en gebruiken.
e) beslisstructuren toepassen in jouw PHP-code.

f) in PHP herhalingslussen maken.
g) invoervelden uitlezen met behulp van PHP.

OPDRACHT 3, VERDIEPING

Deze verdiepingsopdracht maak je als je een stap verder wilt gaan met PHP-scripting of als je al klaar bent met de eindopdracht en het maximaal aantal punten wilt scoren voor deze module. Deze verdieping doet een beroep op jouw kwaliteiten voor *zelfstandig*- en *onderzoekend* leren.

Je gaat voor het formulier dat je gemaakt hebt bij de eindopdracht een zo uitgebreid mogelijke *formulier-validatie* maken. Formulier- of gegevensvalidatie komt voort uit het principe dat je de gegevens die mensen invoeren in jouw formulier nooit mag vertrouwen. Denk hierbij aan het leeg laten van invoervelden, het invullen van letters in velden waar cijfers verwacht worden etc.

Het valideren van gegevens kan voor een deel gedaan worden aan de *voorkant*, in HTML en voor een deel in de *backend*, met bijvoorbeeld PHP. Maar echt goed beveiligen kan pas als je gebruik maakt van *allebei* de manieren.

Op internet is enorm veel informatie te vinden over *form-validation*. Ga zelf maar eens onderzoeken wat er allemaal mogelijk is.

Eisen

- Beveilig jouw formulier, zo uitgebreid mogelijk, met *front- en backendvalidation*.

BEOORDELING

Module	Verdieping Programmeren 1
Naam beoordelaar	
Naam student	
Datum	
Leerdoelen:	
a) De student kan een PHP-pagina maken, opslaan en uploaden naar een webserver --> (wordt beoordeeld in alle opdrachten)	
b) De student kan een PHP-pagina maken met de juiste syntax --> (wordt beoordeeld in alle opdrachten)	
c) De student kan gegevens op het scherm tonen met het echo-commando --> (wordt beoordeeld in alle opdrachten)	
d) De student kan in PHP variabelen declareren, initialiseren en gebruiken --> (wordt beoordeeld in opdracht 2 en 3)	
e) De student kan beslisstructuren toepassen in de PHP-code --> (wordt beoordeeld in opdracht 3)	
f) De student kan in PHP herhalingslussen maken --> (wordt beoordeeld in opdracht 2 en 3)	
g) De student kan invoervelden uitlezen met behulp van PHP --> (wordt beoordeeld in opdracht 3)	
Opmerkingen:	
Normering:	Score:
Leerdoelen voldoende	7
Uitbereiding van de eindopdracht gemaakt	2 (maximaal)
Uitzonderlijk werk	1 (maximaal)
Bij te laat inleveren	*Maximum cijfer: 6*
Niets inleveren	*Cijfer: 1*
Cijfer:	