erasmus

HOGESCHOOL BRUSSEL

Web Development Advanced

erasmus
HOGESCHOOL BRUSSEL

BASISPRINCIPES PHP

Wat is PHP?

- Hypertext Preprocessor
- = een open-source server-side scriptingtaal.
- .php

BASISPRINCIPES PHP

Server-side vs client-side scripting

- Server-side (PHP)
  - Code wordt op de server uitgevoerd
  - Bijvoorbeeld:
    - Genereert een deel HTML code die daarna naar een specifieke client wordt gestuurd
    - Voert taken uit op de server (vb: mails versturen, files wijzigen, databases aanpassen, …)

BASISPRINCIPES PHP

Server-side vs client-side scripting

- Client-side (JavaScript)
  - Code wordt op de client uitgevoerd (browser)
  - Bijvoorbeeld:
    - Image slider

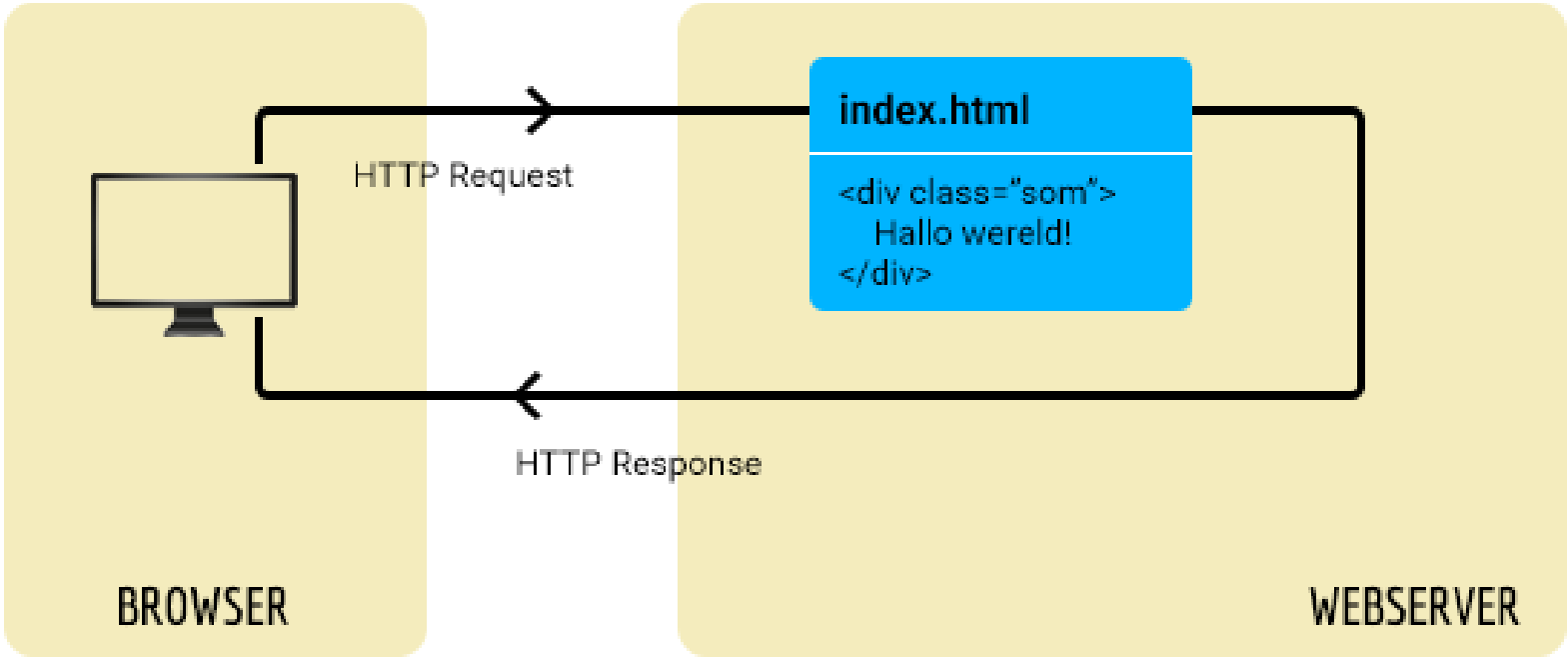
BASISPRINCIPES PHP

Soorten webapplicaties

- Statische webapplicatie
  - HTML, CSS en/of JS bestanden
  - Lokaal/server
- Dynamische webapplicatie
  - HTML, CSS en/of JS bestanden
  - PHP & MySQL
  - Server

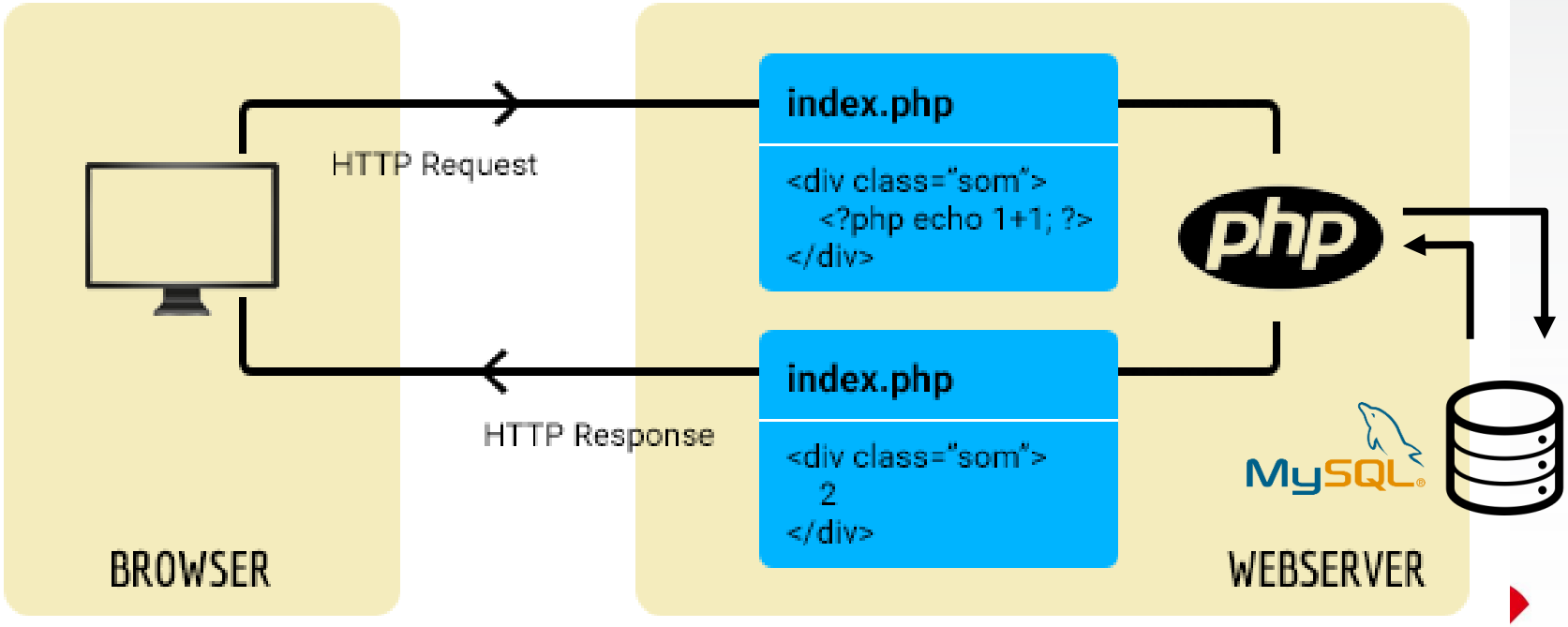
BASISPRINCIPES PHP

Server – statische website



BASISPRINCIPES PHP

Server – dynamische website



erasmus
HOGESCHOOL BRUSSEL

BASISPRINCIPES PHP

HTTP

= Hyper Text Transfer Protocol

- Communicatie tussen browser (client) & server
  - Request
  - Response
- Connectieloos
- Statusloos

BASISPRINCIPES PHP

HTTP - request

- Moet bevatten:
  - Request-line
    - 8 verschillende methoden om actie uit te voeren.
  - Host-header
    - Extra info bv. Domeinnaam server, taal browser, encodering client, cookies, ...

BASISPRINCIPES PHP

HTTP - request

- Kan bevatten:
  - Body
    - Bevat data die meegestuurd wordt naar de server
    - Gebruikt bij POST- en PUT-methode

BASISPRINCIPES PHP

HTTP - request

- Request-line
  - GET
    - Vraagt bepaald bestand op
      bv. HTML bestanden
  - HEAD
    - Gelijk aan GET maar zonder HTTP-body in response

BASISPRINCIPES PHP

HTTP - request

- Request-line
  - POST
    - Data naar server sturen
  - PUT
    - Opslaan van bestanden op server
  - DELETE
    - Verwijderen van bestand(en) op server

BASISPRINCIPES PHP

HTTP - request

- Request-line
  - TRACE
    - Stuurt request terug in de body van de response
  - OPTIONS
    - Vraagt aan server welke header verplicht of optioneel zijn
  - CONNECT
    - Versturen van geëncrypteerde data

BASISPRINCIPES PHP

HTTP - request

http://iwt.ehb.be/pagina/toegepaste-informatica

```
GET /pagina/toegepaste-informatica HTTP/1.1
Host: iwt.ehb.be
Connection: close
User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_6_4;
de-de)
AppleWebKit/533.16 (KHTML, like Gecko) Version/5.0
Safari/533.16
Accept:
text/xml,text/html,text/plain,image/png,image/jpeg,image/gif
Accept-Charset: ISO-8859-1,utf-8
```

BASISPRINCIPES PHP

HTTP - response

- Bevat steeds:
  - Statuscode
  - Headers
    - Info over server
- Kan bevatten:
  - Body

BASISPRINCIPES PHP

HTTP - response

- Statuscode
  - Gegroepeerd in categorieën:
    - 1xx: Mededelend
    - 2xx: Succesvol
    - 3xx: Omleiding
    - 4xx: Clientfout
    - 5xx: Serverfout

BASISPRINCIPES PHP

HTTP - response

```
HTTP/1.1 200 OK
Date: Thu, 01 Sep 2011 08:30:00 GMT
Server: Apache/2.2.16 (Debian) PHP/5.3.3-7+squeeze3 with Suhosin-Patch
X-Powered-By: PHP/5.3.3-7+squeeze3
Set-Cookie:
SESS839c74ae5e365ed4c762b6c3358c3e3c=d3ef6a79fb75b5dad63f45bbd71e24c
5; expires=Wed, 21-Sep-2011 15:54:00 GMT; path=/; domain=iwt.ehb.be
Last-Modified: Thu, 01 Sep 2011 08:29:31 GMT
ETag: "a919476247965d473536b8ac61266c01"
Expires: Sun, 19 Nov 1978 05:00:00 GMT
Cache-Control: must-revalidate
Content-Encoding: gzip
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=utf-8
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="nl" xml:lang="nl">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Departementale website IWT | Industriële Wetenschappen & Technologie
</title>
```

erasmus

HOGESCHOOL BRUSSEL

PHP Synaxis

PHP BESTAND

erasmus
HOGESCHOOL BRUSSEL

PHP SYNTAXIS

PHP bestand

- PHP-code staat tussen HTML code
  - <?php en ?>
  - Alternatief <? En ?>
- HTML code wordt letterlijk overgenomen in resultaat door PHP parser

PHP SYNTAXIS

PHP bestand

- Er hoeft geen HTML code te staan
  - PHP bestand met enkel PHP code is mogelijk
  - Wordt dikwijls gebruikt in praktijk, vb bij toepassing van MVC patroon

PHP SYNTAXIS

PHP bestand - Voorbeeld

```
<!DOCTYPE html>
   <html>
   <head>
      <title>PHP info</title>
   </head>
   <body>
      <?php phpinfo(); ?>
   </body>
</html>
```

PHP BASISSYNTAX

PHP SYNTAXIS

Syntax

- Analoog met Java / C#
- We eindigen onze PHP-coderegels met een puntkomma

PHP SYNTAXIS

Witruimte

- PHP is “ongevoelig” voor witruimte in code
- Uiteraard wel om scheiding tussen woorden aan te duiden
- Hoeveelheid witruimte maakt niet uit (maar niet overdrijven!)

PHP SYNTAXIS

Commentaar

- 2 soorten
  - Eenregelige commentaar
    - Aanduiding: // voor code
    - Alternatief: # niet gebruiken
  - Commentaar blok
    - Aanduiding: /* code */

PHP SYNTAXIS

Commentaar - voorbeeld

```
<?php
  //eenregelige commentaar

  /*
  meerdere regels
  in een commentaarblok
  */
?>
```

PHP SYNTAXIS

Hoofdlettergevoeligheid

- Wel
  - Variabelenamen
  - Sleutelwoorden
    - if, then, else, for, while, …
- Niet
  - Functienamen
- Afspraak: ordelijk programmeren impliceert dat we ervan uitgaan dat alles case sensitive is

VARIABELEN

erasmus
HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Variabelen

- Naam toewijzen aan een bepaalde waarde
- Voorafgegaan door $
- Toewijzing door = operator
- Meest recente toewijzing is van kracht

PHP SYNTAXIS

Variabelen

- Mogelijk om te declareren voordat ze gebruikt worden
- Hebben zelf geen type. Type wordt bepaald door de waarde
- Hebben een standaardwaarde
- Initialisatie kan overal in script

PHP SYNTAXIS

Variabelen - naamgeving

- Aanduiding door middel van een $
- Eerste teken moet een letter zijn
- Daarna mogen
  - Letters
  - Cijfers
  - Underscore: _
- Gebruikelijk om variabelenamen in camelCase te schrijven

PHP SYNTAXIS

Variabelen - voorbeeld

```
<?php
    $auteur = “Shakespeare";

    //expressie in variabele
    $pi = 3 + 0.14159;

    //Variabele heeft geen type
    //PHP is “zwak getypeerde taal”
    $pi = "3 + 0.14159";

    //Naamgeving in camelCase
    $langeVariabeleNaam = “test";
?>
```

PHP SYNTAXIS

Variabelen – automatische casting

- PHP is zwak getypeerde taal en ondersteund automatische casting
- Casting = omzetten naar een ander type
- Heeft voor- en nadelen
  - Er worden geen fouten gegenereerd => Moeilijk om fouten die veroorzaakt worden door automatische casting op te sporen

PHP SYNTAXIS

Variabelen – automatische casting

```
<?php
   $var1 = ”12 uur"; //string
   $var2 = 25; //integer
   $var3 = $var1 + $var2; // 37
?>
```

PHP SYNTAXIS

Variabelen – automatische casting

```
<?php
   $var1 = ”uur 12"; //string
   $var2 = 25; //integer
   $var3 = $var1 + $var2; // 25
?>
```

PHP SYNTAXIS

Variabelen – niet toegewezen

```
<?php
   $var1 = "test";

   //drukt 'true' af op het scherm
   echo isset($var1);

   //drukt 'false' af op het scherm
   echo isset($var2);
?>
```

PHP SYNTAXIS

Variabelen – soorten

- Globaal
  - Bevindt zich niet in een functie of klasse
  - Beschikbaar in andere PHP-bestanden
  - Enkel globaal aan te spreken
    - Binnen functie: global

PHP SYNTAXIS

Variabelen – soorten

- Lokaal
  - Bevindt zich binnen een functie
  - Bij verlaten functie: wordt verwijdert

PHP SYNTAXIS

Variabelen – globaal vs lokaal

```
<?php
   $a = 3; // GLOBAAL
   globaleVariabelen();
   function globaleVariabelen() {
      $a = 4; // LOKAAL
      print $a; // 4

      global $a;
      print $a; // 3
   }
?>
```

PHP SYNTAXIS

Variabelen – soorten

- Statisch
  - Bevindt zich binnen een functie
  - Bij verlaten functie: behoudt waarde

PHP SYNTAXIS

Variabelen – static

```
<?php
   function statischeVariabelen($b) {
      static $a;
      $a += $b;
      return $a;
   }
   print statischeVariabelen(4); // 4
   print statischeVariabelen(6); // 10
?>
```

PHP SYNTAXIS

Variabelen – soorten

- Constanten
  - Toekennen van bepaalde waarde aan bepaalde naam
    - define(constante, waarde)
  - Variabelen die éénmalig worden geïnitialiseerd
  - In hoofdletters
  - Zowel globaal als lokaal beschikbaar

PHP SYNTAXIS

Variabelen – constanten

```
<?php
   //definieren
   define("PI",3.14159);
?>

<?php
   //Gebruik
   $var = PI;
?>
```

PHP SYNTAXIS

Datatypen - eenvoudige

- Integers
  - De gehele getallen.
- Doubles
  - Floating-point getallen of decimale getallen.
- Booleans
  - Datatype dat twee waarden kent: true of false.

PHP SYNTAXIS

Datatypen - eenvoudige

- NULL
  - Een speciaal type dat slechts één waarde heeft: null.
- Strings
  - Tekenreeksen.

PHP SYNTAXIS

Datatypen - samengestelde

- Arrays
  - Benoemde of geïndexeerde verzameling van andere waarden.
- Objecten
  - Instanties van een bepaalde klasse

PHP SYNTAXIS

Datatypen - samengestelde

- Resources
  - Speciale variabelen die een verwijzing bevatten naar externe bronnen (buiten PHP) zoals een databaseverbinding bijvoorbeeld.

FUNCTIES

PHP SYNTAXIS

Functies

- Moeten ergens in het script éénmaal gedeclareerd worden om ze te kunnen aanroepen
- Kunnen ook binnen andere functies worden aangeroepen
- Er is geen main functie, het script begint aan het begin van de pagina
- Functienamen
  - niet case sensitive
  - Afspraak: schrijf in camelCase

PHP SYNTAXIS

Functies - voorbeeld

```
function functienaam($arg1, $arg2, ...)
{
   //functie invulling
   return “resultaat”;
}
```

PHP SYNTAXIS

Functies - argumenten

- Functie wordt aangeroepen met te weinig argumenten
  - Functie wordt aangeroepen
  - Warning wordt gegenereerd (dikwijls niet zichtbaar)
  - Ontbrekende argumenten worden behandeld als ongedefinieerde variabelen

PHP SYNTAXIS

Functies - argumenten

- Functie wordt aangeroepen met te veel argumenten
  - Functie wordt aangeroepen
  - Extra argumenten worden genegeerd

PHP SYNTAXIS

Functies – standaard functies

- PHP installatie komt met een bibliotheek van standaardfuncties waar we ten alle tijde gebruik van kunnen maken.

PHP SYNTAXIS

Functies – echo

- Uitvoer van tekst naar het resultaat van het PHP script (de HTML pagina die wordt samengesteld)

```
<?php
    echo("Een string afdrukken");
?>
```

PHP SYNTAXIS

Functies – echo

- De echo functie wordt zo veel gebruikt dat zij zonder haakjes mag worden geschreven
  - Dit is een uitzondering
  - Normaal roepen we functies aan met haakjes !!!

```
<?php
 echo "Een string afdrukken";
?>
```

PHP SYNTAXIS

Functies – echo

- Geen overmatig gebruik !
- Echo wordt gebruikt om de waarden van variabelen naar de client te outputten
- Het is niet de bedoeling om al te veel statische HTML code te outputten naar de client.
  - Sluit in dat geval het PHP blok even af en schrijf je statische HTML code, open daarna opnieuw een PHP blok voor de rest van je PHP code

CONTROLE-STRUCTUREN

PHP SYNTAXIS

Controlestructuren

- Onderdelen
  - Testexpressie
    - booleanse expressie (TRUE of FALSE)
    - Precedentieregels
  - Uit te voeren code indien testexpressie = true

PHP SYNTAXIS

Rekenkundige operatoren

Operator	Verklaring
+	Optelling
-	Aftrekking
*	Vermenigvuldiging
/	Deling
%	Rest

PHP SYNTAXIS

Rekenkundige operatoren

Operator	Verklaring
$a++	$a = $a + 1
$a--	$a = $a - 1
$a += $b	$a = $a + $b
$a -= $b	$a = $a - $b
$a *= *b	$a = $a * $b
$a /= $b	$a = $a / $b

PHP SYNTAXIS

Vergelijkingsoperatoren

Operator	Verklaring
==	Is gelijk aan
<	Kleiner dan
>	Groter dan
<=	Kleinder dan of gelijk aan
>=	groter dan of gelijk aan
!= / <>	Verschillend van
===	Gelijk aan en van hetzelfde type

PHP SYNTAXIS

Logische operatoren

Operator	Verklaring
&& / and	Logische and
|| / or	Logische or
Xor	Exclusieve or (iderm or, geeft false indien 2x true)
!	not

PHP SYNTAXIS

Controlestructuren - precentieregels

- Basis wikunde
- Zie: http://php.net/manual/en/language.operators.precedence.php
- Belangrijkste: vergelijkingsoperatoren hebben voorrang op logische operatoren

PHP SYNTAXIS

Controlestructuren - precentieregels

```
if($var1 == $var2 && $var2 > $var3)
{
      return true;
}
```

PHP SYNTAXIS

Controlestructuren – if/else

```
if($var1 <= $var2) {
   // waar
   echo "Dit is waar";
} else {
   // niet waar
   echo "Dit is niet waar";
}
```

PHP SYNTAXIS

Controlestructuren – if/else

- Verkorte structuur
  - Conditie ? indientrue : indienfalse

```
($var1 <= $var2)? ”var2 grootste” : ”var1 grootste”;
```

PHP SYNTAXIS

Controlestructuren – switch/case

- Evalueert een bepaalde variabele, en ziet of deze gelijk is aan de waarde die bij een bepaalde "case" wordt gegeven
- Schrijf op het einde van elke case een "break;" commando
- Mogelijkheid om een default blok te maken: Deze code wordt uitgevoerd wanneer er geen code in een ander case blok werd uitgevoerd

PHP SYNTAXIS

Controlestructuren – switch/case

```
<?php
   switch ($i) {
      case 0:
         echo "i is gelijk aan 0";
         break;
      case 1:
         echo "i is gelijk aan 1";
         break;
      default:
         echo "i is niet gelijk aan 0, 1 of 2“;
   }
?>
```

erasmus HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Lussen - While

- Voorwaarde wordt nagekeken voor starten lus-code
- Lus-code wordt herhaald zolang voorwaarde true is

```
<?php
    $i = 1;
    while ($i <= 10) {
        echo $i++;
    }
?>
```

PHP SYNTAXIS

Lussen – Do/While

- Voorwaarde wordt nagekeken na het uitvoeren van de lus-code
- De lus-code wordt dus altijd minstens 1 maal uitgevoerd
- Lus-code wordt herhaald zolang voorwaarde true is

PHP SYNTAXIS

Lussen – Do/While

```
<?php
   $i = 0;
   do {
      echo $i;
   } while ($i > 0);
?>
```

PHP SYNTAXIS

Lussen – For

- Opbouw

```
for(expressie1, expressie2, expressie3) {
    opdrachten
}
```

- Expressie1: 1x uitgevoerd bij begin loop
- Bij begin van elke iteratie wordt expressie2 geëvalueerd => iteratie wordt voortgezet wanneer expressie2=true
- Op het einde van elke iteratie wordt expressie3 uitgevoerd

erasmus HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Lussen – For

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

PHP SYNTAXIS

Controlestructuren en HTML

- Je kan HTML code mixen doorheen PHP code en controlestructuren
- Maak hier gebruik van !
- Te veel statische HTML code outputten door middel van echo is niet gewenst.

PHP SYNTAXIS

Controlestructuren en HTML

```
<?php if($webbrowser=="ie") { ?>
      <link rel="stylesheet" type="text/css"
            href="style_ie.css”/>
<?php
} else {
?>
      <link rel="stylesheet" type="text/css"
            href="style.css"/>
<?php } ?>
```

erasmus
HOGESCHOOL BRUSSEL

BASISFUNCTIES

erasmus
HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Ingebouwde functionaliteiten

- Ouput
  - Echo
  - Print
- Phpinfo()
- Isset()
- Empty()
- Include
- Require
- Die
- Header-redirect
- datumfuncties

PHP SYNTAXIS

Output

- Uitvoer van tekst naar het resultaat van het PHP script (de HTML pagina die wordt samengesteld)
- Optie 1: echo

```
<?php
   echo "Een string afdrukken”;
   echo("Een string afdrukken");
?>
```

PHP SYNTAXIS

Print

- Tweede optie voor output te maken
- Slechts één argument
- Geeft return waarde
  - Geeft weer of print gelukt (=1) of mislukt (=0) is
- Normaal gebruiken we “echo”

PHP SYNTAXIS

phpinfo()

- Geeft nuttige informatie over de status van de PHP installatie
- Wordt gebruikt om de configuratie van de PHP te checken
  - Vb: informatie over
  - Geïnstalleerde extensies
  - Versie
  - Omgevingsvariabelen
  - …

PHP SYNTAXIS

isset()

- De isset-functie geeft een boolean terug die aangeeft of er reeds een waarde werd toegekend aan een variabele, en of die waarde niet NULL is

```
<?php
    $var1 = "test";
    echo isset($var1);
    //drukt 'true' af op het scherm
    echo isset($var2);
    //drukt 'false' af op het scherm
?>
```

PHP SYNTAXIS

empty()

- Geeft een boolean terug die weergeeft of een variabele wordt beschouwd als leeg.
- Een variabele wordt als leeg beschouwd als deze niet bestaat of als zijn value gelijk is aan één van de volgende waarden:
  - “”, 0, “0”, NULL, false, een lege array

PHP SYNTAXIS

empty()

- Geeft een boolean terug die weergeeft of een variabele wordt beschouwd als leeg.
- Een variabele wordt als leeg beschouwd als deze niet bestaat of als zijn value gelijk is aan één van de volgende waarden:
  - “”, 0, “0”, NULL, false, een lege array

erasmus HOGESCHOOL BRUSSEL

php

PHP SYNTAXIS

empty() – isset()

```
<?php
    $var = 0;
    if (empty($var)) {
        //wordt uitgevoerd
        echo '$var is empty';
    }
    if (isset($var)) {
        //wordt uitgevoerd ondanks dat var empty is
        echo '$var is set';
    }
?>
```

erasmus
HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Include en require

- Je kan in PHP andere bestanden toevoegen
  - Vb: apart php bestand met een bibliotheek van functies
- Bestanden die worden toegevoegd worden geïnterpreteerd als HTML
  - Daarom terug <?php en ?> invoegen in deze files als je van PHP gebruik wilt maken

PHP SYNTAXIS

Include en require

- 4 mogelijkheden
  - Include
  - Require
  - Include_once
  - Require_once

PHP SYNTAXIS

Include

- Wanneer het bestand niet gevonden wordt => warning
- Vindt enkel plaats als het statement wordt uitgevoerd

PHP SYNTAXIS

Include

```
include(“bestand1.php”);
include “bestand2.php”;

if ($test) {
   include(“bestand3.php”);
   //include wordt enkel toegevoegd
   //wanneer $test = true
}

include “bestand4.php”;
```

PHP SYNTAXIS

Require

- Wanneer het bestand niet gevonden wordt => error
- Wordt altijd uitgevoerd

PHP SYNTAXIS

Require

```
require(“bestand1.php”);
if ($test){
  require(“bestand3.php”);
  // bestand3 wordt altijd
  // toegevoegd, ook wanneer
  // wanneer $test = false
}
```

PHP SYNTAXIS

Include_once, require_once

- Indien een bepaald bestand 2 keer wordt geïnclude, zal PHP protesteren, omdat hij bijvoorbeeld 2 definities van dezelfde functie terugvindt

- Gebruik bovenstaande methodes om dit te vermijden. PHP zal zelf bijhouden of de files in het verleden reeds werden toegevoegd, en dit geen tweede keer toevoegen.

PHP SYNTAXIS

die

- Om de uitvoering van je PHP script te stoppen
- Probeer uitvoering van *die* te vermijden in een productieomgeving
  - Situatie opvangen en een mooie melding weer geven naar de gebruiker
  - Eventueel gebruiker doorverwijzen naar een andere pagina

PHP SYNTAXIS

Datum & tijd

- time()
  - Geeft het aantal seconden terug, gemeten sinds 1 januari 1970 om 00u00
  - Je kan gemakkelijk rekenen met deze tijd door hier een hoeveelheid seconden bij op te tellen

PHP SYNTAXIS

Datum & tijd

```
<?php
   $nu = time();
   $volgendeWeek = time() + (7 * 24 * 60 * 60);
   // 7 dagen, 24 uur, 60 minuten, 60 sec
?>
```

PHP SYNTAXIS

Datum & tijd

- date()
  - Geeft een gegeven tijdstip geformatteerd volgens een gegeven format string
  - 2 parameters

PHP SYNTAXIS

Datum & tijd

- date()
  - 2 parameters
    - Format string
      - Het formaat van de uitvoer (string) van deze functie wordt door dit argument bepaald
      - Mogelijkheden terug te vinden op: http://php.net/manual/en/function.date.php
    - Timestamp
      - Optioneel
      - Indien niet gegeven: gelijk aan time()
      - In seconden sinds 1 januari 1970 om 00u00

PHP SYNTAXIS

Datum & tijd

```
<?php
    $nu = time();
    $volgendeWeek = time() + (7 * 24 * 60 * 60);
    echo 'Nu: '. date('d-m-Y', $nu);
    echo 'Nu: '. date('d-m-Y');
    echo 'Volgende week: '. date('Y-m-d', $volgendeWeek);
    //Dit script toont iets als:
    //14-02-2017
    //14-02-2017
    //21-02-2017
?>
```

PHP SYNTAXIS

Datum & tijd

- mktime()
  - We kunnen ook gebruik maken van deze alternatieve functie om gemakkelijk een tijdstip in te stellen
  - Geeft een integer terug die het opgegeven tijdstip representeert in seconden, gerekend vanaf 1 januari 1970.

PHP SYNTAXIS

Datum & tijd

- mktime()
  - mktime(<uur>,<minuut>,<seconde>,<maand>,<dag>,<jaar>);

```
<?php
   //tijdtip 31/12/2017 00:00:00
   $datum = mktime(0,0,0,12,31,2017);
?>
```

STRINGS

PHP SYNTAXIS

Strings

- “tekst” of ‘tekst’
- Speciale tekens escapen door middel van backslash

```
<?php
   $var1 = “Acteur: \“Test\””;
   $var2 = ‘\’s Morgens groet Mark de
   dingen’;
?>
```

PHP SYNTAXIS

Strings

- Alles wat tussen dubbele aanhalingstekens staat wordt geëvalueerd door de PHP parser

```
<?php
   $woord = "vakantie";
   echo 'Ik wil $woord';
   //uitvoer: Ik wil $woord
   echo "Ik wil $woord";
   //uitvoer: Ik wil vakantie
?>
```

PHP SYNTAXIS

Strings - concatenatie

- Aaneen plakken van meerdere strings
- Gebruik hiervoor een punt .

```
<?php
   $var1 = “eerste “.“tweede “; // eerste tweede
   $var2 = “derde";
   $var3 = $var1 . $var2; //eerste tweede derde
   $var1 .= $var2; //eerste tweede derde
?>
```

erasmus HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Strings - functies

- Je kan strings gemakkelijk manipuleren met ingebouwde stringfuncties
- Zie manual voor volledig overzicht:

  vb: http://be1.php.net/manual/en/function.trim.php

PHP SYNTAXIS

Strings - functies

- strpos(string1, string2)
  - geeft de numerische positie van het eerste voorkomen van string 2 in string 1 terug
- strtolower(string)
  - zet de string om naar kleine letters
- strtoupper(string)
  - zet de string om naar hoofdletters

PHP SYNTAXIS

Strings - functies

- strlen(string)
  - geeft de lengte van de string terug
- string wordwrap(input, aantal kol, nextlinechar)
  - zet string om naar verschillende lijnen
- strcmp(string1, string2)
  - returnt true indien zelfde string

PHP SYNTAXIS

Strings - functies

- strncmp(string1, string2, n)
  - returnt true indien zelfde eerste n karakters
- ucfirst(string)
  - het eerste karakter van de string wordt een hoofdletter
- ucwords(string)
  - het eerste karakter van elk woord in de string wordt met een hoofdletter geschreven

PHP SYNTAXIS

Strings - functies

- substr(string, startpos, lengte)
  - geeft een deel van string terug
- trim(string)
  - geeft de inputstring terug, met alle whitespace aan het begin en einde ervan verwijderd
  - variaties mogelijk waar je een alternatief geeft voor whitespace
  - andere variaties ltrim en rtrim

PHP SYNTAXIS

Strings - functies

- addslashes(string)
  - zet tekens om naar hun escaped equivalent wanneer nodig
- stripslashes(string)
  - maakt addslashes ongedaan
- htmlspecialchars(string)
  - karakters die in html een speciale betekenis hebben vervangen door hun equivalent
  - voorbeeld: & wordt &
- $var[5]
  - haalt een bepaald karakter van een string op

erasmus HOGESCHOOL BRUSSEL

ARRAYS

PHP SYNTAXIS

Arrays

- Een verzameling/collectie van allerlei gegevens
- 2 soorten
  - Geïndexeerde array
  - Associatieve array

PHP SYNTAXIS

Arrays - geïndexeerd

- Gebruikt een index als sleutel om een item in de collectie te identificeren
- De index is een integer
- Zero based
- 2 manieren om aan te maken

PHP SYNTAXIS

Arrays - geïndexeerd

```
<?php
	$array[0] = "element 1";
?>
```

Of

```
<?php
	$array = array("element 1","element2");
?>
```

PHP SYNTAXIS

Arrays - Associatieve

- De sleutel om een item in de collectie te identificeren is geen index-waarde, maar een string
- De key is case sensitive
- 2 manieren van aanmaak

PHP SYNTAXIS

Arrays - Associatieve

```
<?php
   $array1["key1"] = "element 1";
   $array1["key2"] = "element 2";
   $array2 = array(
      "key1" => "element 1",
      "key2" => "element 2“
   );
?>
```

erasmus
HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Arrays - overlopen

- Eventueel met een for/while lus
  - Gebruik hiervoor de count(array) functie
- Beter: foreach lus
  - Bij deze lus zal elke waarde in een bepaalde array overlopen worden

PHP SYNTAXIS

Arrays - overlopen

```
<?php
   $arr[0] = 4352;
   $arr[1] = 342;
   $arr[2] = 42;
   $arr[3] = 65875;
   for($i = 0; $i < count($arr); $i++) {
      echo $arr[i];
   }
?>
```

PHP SYNTAXIS

Arrays - overlopen

```
<?php
    $array = array("achternaam"=>"Vermeulen",
    "email"=>"joske.vermeulen@gmail.com");
    foreach($array as $sleutel => $waarde) {
        echo "sleutel = ". $sleutel ."<br />waarde = ".
        $waarde . "<br /><br />";
    }
?>
//sleutel = achternaam
//waarde = Vermeulen
//sleutel = email
//waarde = joske.vermeulen@gmail.com
```

erasmus HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Arrays - overlopen

```
<?php
    $array = array(
        "achternaam"=>"Vermeulen",
        "email"=> "joske.vermeulen@gmail.com"
    );
    foreach($array as $waarde) {
        echo "waarde = ". $waarde . "<br />";
    }
?>
//waarde = Vermeulen
//waarde = joske.vermeulen@gmail.com
```

erasmus HOGESCHOOL BRUSSEL

PHP SYNTAXIS

Arrays - functies

- Hele lijst beschikbaar op: http://be2.php.net/manual/en/book.array.php
- Voorbeelden:
  - array_key_exists
    - gaat na of de key bestaat in de array
  - array_keys
    - geeft een lijst terug van alle keys
  - array_merge
    - voegt arrays samen

erasmus HOGESCHOOL BRUSSEL

php

PHP SYNTAXIS

Arrays - functies

- count
  - telt alle elementen in een array
- sort
  - sorteert een array, items kunnen nieuwe index toegewezen krijgen
- asort
  - sorteert een array, items behouden hun index (belangrijk bij bijvoorbeeld associatieve array)
- shuffle
  - schudt de volgorde van de items in een array door elkaar

PHP SYNTAXIS

Arrays - multidimensionaal

- Een array in een array
- Als je alle elementen wilt overlopen, moet je dus controlestructuren nesten

PHP SYNTAXIS

Arrays - multidimensionaal

```
<?php
    $score[“Ruud”][“PHP”] = 16;
    $score[“Ruud”][“AS3”] = 20;
    $score[“Ruud”][“Project”]=16;

    $score[“Jan”][“PHP”] = 15;
    $score[“Jan”][“AS3”] = 19;
    $score[“Jan”][“Project”]=16;

    $score[“Alex”][“PHP”] = 14;
    $score[“Alex”][“AS3”] = 11;
    $score[“Alex”][“Project”]=14;
?>
```

erasmus HOGESCHOOL BRUSSEL

HTTP HEADER

PHP SYNTAXIS

Header functie

- Laat toe om de headers te veranderen die met de HTTP response worden teruggestuurd
- Je moet dit doen als eerste regel op je PHP pagina, anders werkt het niet !
- Ook geen spaties of lege regels voor de commando's zetten
- Bijvoorbeeld: redirect naar andere pagina

PHP SYNTAXIS

Header functie - redirect

- Je kan een gebruiker naar een andere webpagina doorsturen
- Je past hiervoor de HTTP header aan

```
<?php
        header("location:http://www.google.be");
?>
```