Einführung in die Webtechnologien

Dipl.-Inf., Dipl.-Ing. (FH) Michael Wilhelm

Hochschule Harz

FB Automatisierung und Informatik

mwilhelm@hs-harz.de

Raum 2.202

Tel. 03943 / 659 338

Inhalt

1. HTML und CSS
2. Formulare
3. **PHP**
4. PHP-OOP
5. PHP-Datenbanken
6. Unix
7. Unix Shellprogrammierung

Dynamische Webseiten mit HTML, PHP, MySQL

- Ziel von PHP
  - Formularauswertung
  - Dateibearbeitung (Bilder)
  - Funktionen für das Upload
  - Anbindung an Datenbanken
- Aufbau der Sprache
  - Prozedural
  - Objektorientiert
- Was braucht man um mit PHP zu arbeiten?
  - Linux oder Windows, Apache, PHP,
  - Datenbank MySQL
  - LAMP oder WAMP
- Beispiele

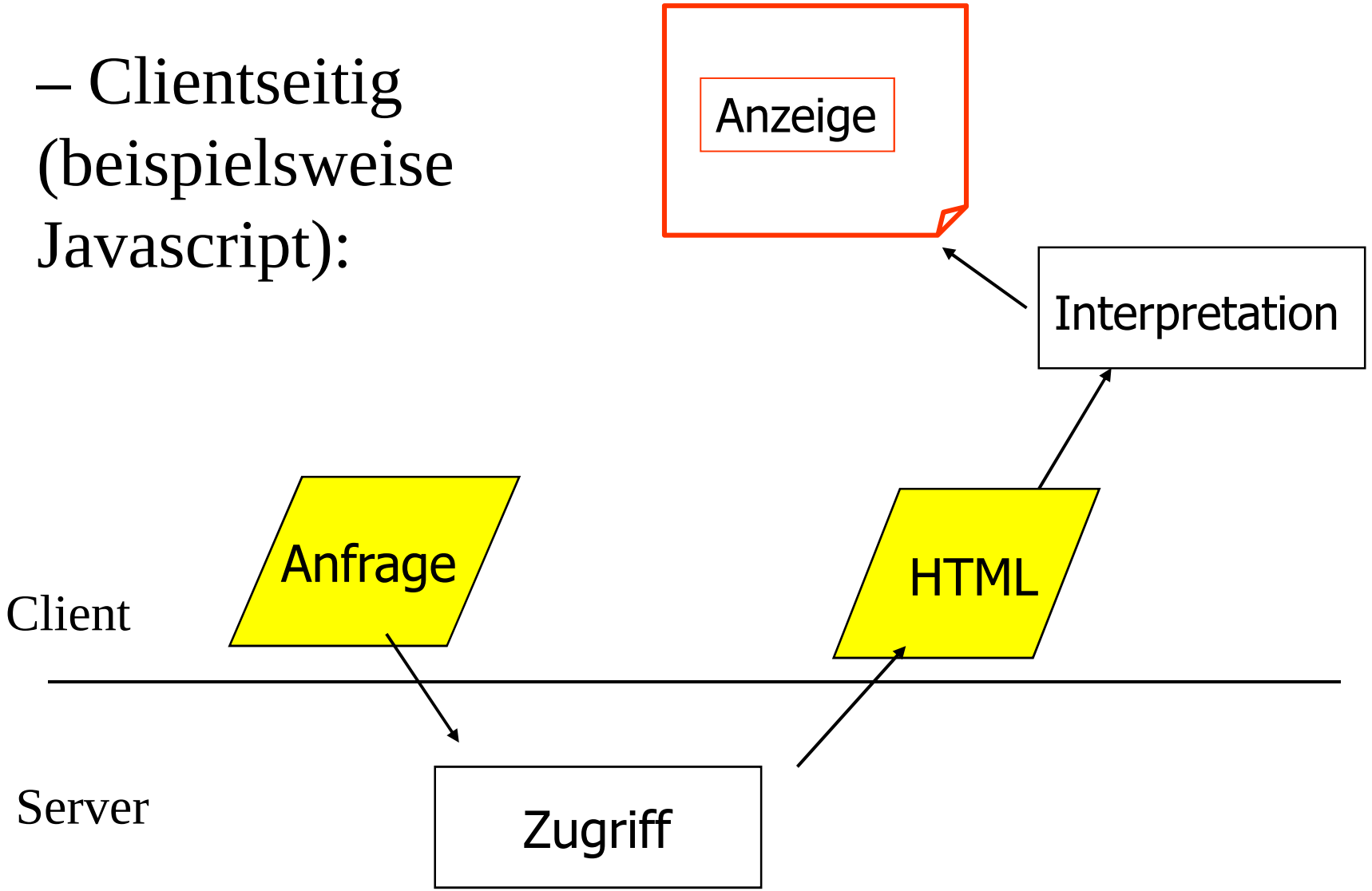
PHP

Was ist PHP?

- PHP: Hypertext Preprocessor
- Serverseitige Skriptsprache
- Syntax ähnlich C, Perl, Java, …
- Eingebettet in HTML
- Ermöglicht dynamische Webseiten
- Anbindung an Datenbanken (z.B. MySQL)
- hat globale, statische und dynamische Variable
- hat Konstanten, Funktionen
- hat Klassen

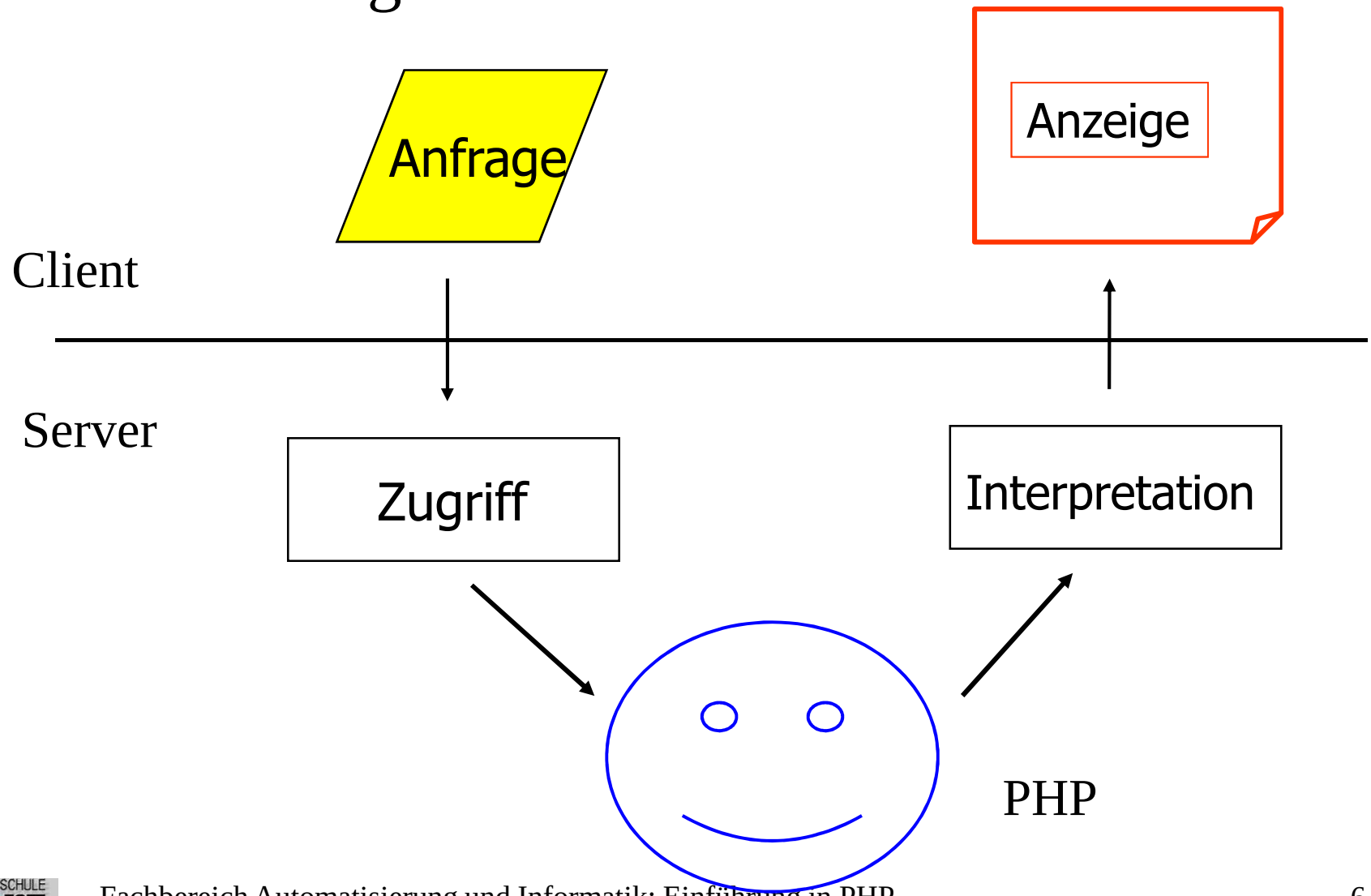
Clientseitige vs. Serverseitige Skriptsprache

– Clientseitig (beispielsweise Javascript):



Clientseitige vs. Serverseitige Skriptsprache

– Serverseitig



PHP

- PHP/FI 1.0, 1995, PHP - "Personal Home Page Tools"
- PHP/FI 2.0, Sommer 1995, FI - "Form Interface„ noch ohne echten Parser
- PHP 3.0, 1997-2010, "Personal Home Page" oder "PHP HyperText Preprocessor"
  - echter Parser
  - Interpreter
  - grundlegende OO Notation
  - people hate Perl
- PHP 4.0, Frühling 2000
  - neuer, schnellerer Sprachkern "Zend"
  - viele, neue Funktionen v.a. bei den Arrays
  - in der Praxis: 2-5x schneller, im Einzelfall: 100x.
  - Compiler ("Encoder"), Debugger, Zend-Cache und IDE in Reichweite
- PHP 5.0, ab 2005
  - objektorientiert
  - Eclipse PlugIn
  - Netbeans Version

Einbindung von PHP in HTML

- PHP-Code wird durch einen Interpreter ausgeführt.

Codierung:

- <?php code ?> (XML)
- <? code ?> (Historisch)
- <% code %> (Historisch, ASP-Tags)
- <script language="php"> code </script>

- **include("conf/conf.php");**

Einbindung von PHP in HTML

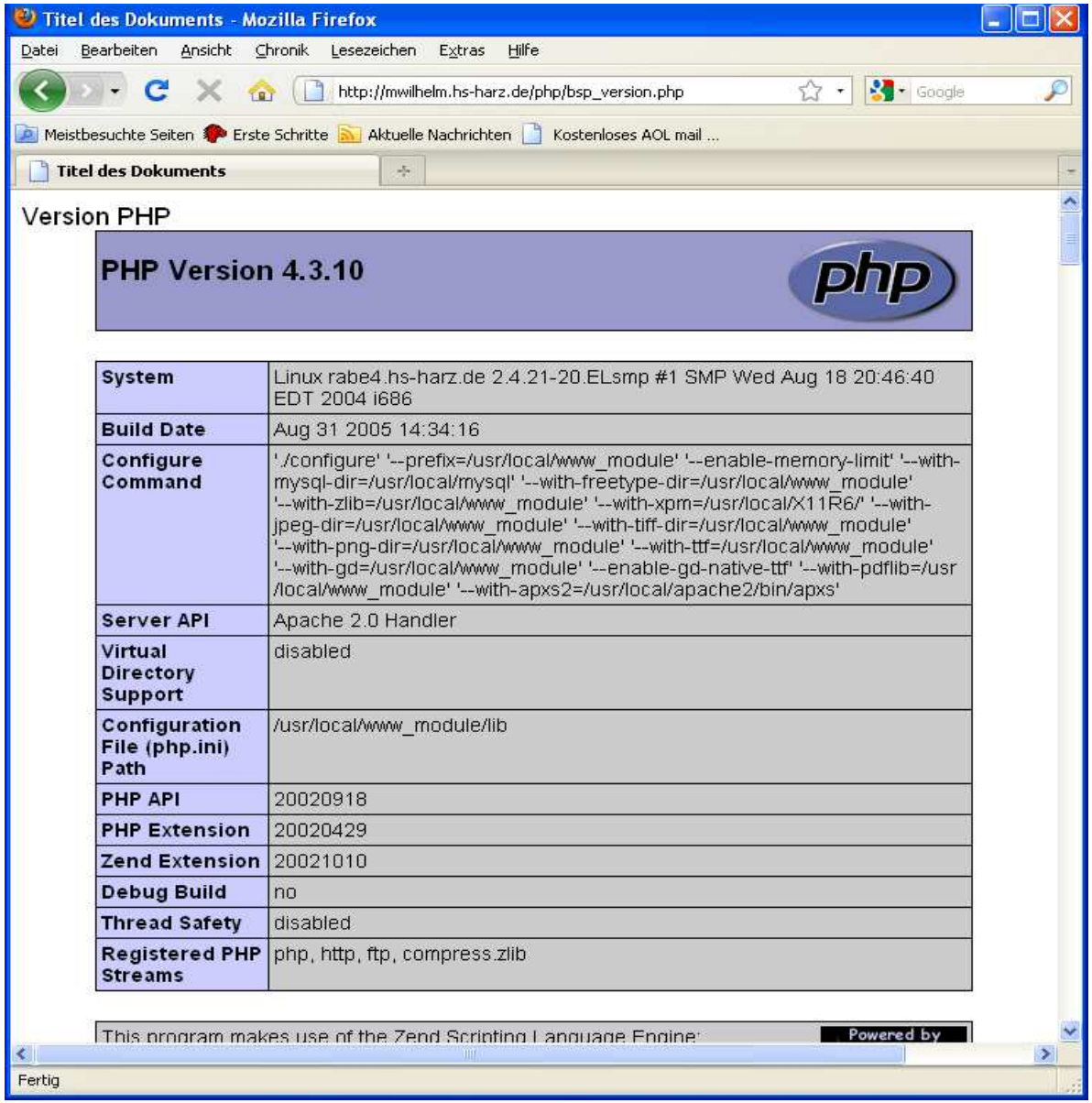
Eigenschaften von PHP-Code

- Variablen
- Kontrollstrukturen (if, case)
- Schleifen (for, while)
- Datentypen (Integer, Double, Bool, Strings)
- Arrays (einfache, multidimensionale, indizierte, assozierte)
- Funktionen
- Bild-Erstellung
- OOP, ab Version 5.0
- Anbindung an eine Datenbank
- Dateioperationen
- Dateiendungen (php, phtml)

1. Beispiel

```
<html>
      <head>
        <title> Titel des Dokuments </title>
      </head>
<body>
   <!--  http://mwilhelm.hs-harz.de/scripte/php/bsp01.php  -->
   <h2>  Erste PHP-Datei  </h2>
   <?php
    echo("hallo Text");
    ?>
   <p>
    HTML-Kern des Dokuments
   </p>
</body>
</html>
```

PHP Version: phpinfo();



Grundlagen von PHP

- Befehle werden durch ein Semikolon abgeschlossen
- ein fehlendes Semikolon führt zu einer Fehlermeldung

- Variablen werden durch ein $ referenziert (auch links vom =)!

Kommentare:

- machen den Code für andere lesbar
- Mittelweg: nicht zu viel, nicht zu wenig
- in PHP durch zwei Schrägstriche: //
- mehrzeilige Kommentare mit /* Kommentar */

Lesbarkeit von PHP

- Guter Code zeichnet sich vor allem dadurch aus, dass er lesbar ist. Daher sind einige Grundregeln zu beachten.

- Ein Beispiel:

```
$a=100;$b=3;$a*=((100-$b)/100);function doit($a,$b=10){
if($b==7){return $a*(1+(7/100));}else{
return $a*(1+(19/100));}}$a=doit($a,7);echo($a);
```

- Was macht dieses Programm?

Lesbarkeit von PHP

Ein Beispiel:

```
$a=100;
$b=3;
$a=$a * ((100-$b)/100);
function doit($a,$b=19){
        if($b==7){
            return $a*(1+(7/100));
        }else{
            return $a*(1+(19/100));
        }
    }
$a=doit($a,7);
echo($a);          103.79
```

Lesbarkeit von PHP

```
$a=100;
$b=3;
$a=$a * ((100-$b)/100);
function doit($a,$b=19){
        if($b==7){
            return $a*(1+(7/100));
        }else{
            if($b==19)
               return $a*(1+(19/100));
            else
              ERROR
        }
}
$a=doit($a,7);
echo($a);          103.79
```

Typen, Variablen, Kontrollstrukturen, Operatoren

Datentypen

Einfache Datentypen:

- Boolean (Werte: TRUE und FALSE)
- Integer (Ganze Zahlen)
- Float, Double (Fließkommazahlen)
- String (Zeichenketten)

Zusammengesetzte Datentypen:

- Array
- Klassen / Objekte

Typen, Variablen, Kontrollstrukturen, Operatoren

Variablen

- Speicherplatz für einen Wert
- Kennzeichnung: "$"+ Name
- Beispiel: $vorname
- Groß- / Kleinschreibung wird unterschieden
- Zuweisung: $i = 10;
- Ausgabe: echo $i;
- $a = 5; *// legt einen Integer an, gettype() = int*
- $a = 5.2; *// legt eine Fließkommazahl an, gettype =double*
- $a = "Heute ist Montag"; *// gettype()=string*

Aufgabe: Ausgabe der Datentypen

Name: bsp_gettype.php

```
 $zahl = 5;
  $str = "3.1415";
  $bool = true;
```

```
<?php
  $zahl = 5;
  echo gettype($zahl);
?>
```

HTML-Listen durch PHP

```
<?php
  echo "<ul>" ;
  echo "  <li>  Grundlagen der Informatik </li>" ;
  echo "  <li>  Mathematik  </li>" ;
  echo "  <li>  Physik  </li>" ;
  echo "  <li>  Englisch </li>" ;
  echo "</ul>" ;
?>
```

http://mwilhelm.hs-harz.de/scripte/php/bsp02.php

HTML-Listen durch PHP

```
<?php
  echo '<ol>' ;
  echo "  <li>  Grundlagen der Informatik </li>" ;
  echo "  <li>  Mathematik  </li>" ;
  echo "  <li>  Physik  </li>" ;
  echo "  <li>  Englisch </li>" ;
  echo "</ol>" ;
?>
```

http://mwilhelm.hs-harz.de/scripte/php/bsp02a.php

HTML-Tabellen durch PHP

```
<?php   echo "<table>" ;
        echo "<tr>";
        echo "<td>  1. Zelle </td>";
        echo "<td>  2. Zelle </td>";
        echo "</tr>";
        echo "<tr>";
        echo '<td colspan="2"  align="center">  3. Zelle </td>';
        echo "</tr>";
        echo "<tr>";
        echo "<td>  3. Zelle </td>";
        echo "<td>  3. Zelle </td>";
        echo "</tr>";
        echo "</table>";
    ?>
```

http://mwilhelm.hs-harz.de/php/bsp03.php

HTML-Tabellen durch PHP

```
<?php   echo "<table>" ;
        for ($i=0;$i<5;$i++) {
        echo "<tr>";
        echo "<td>  $i 1. Zelle </td>";
        echo "<td>  2. Zelle </td>";
        echo "</tr>";
        echo "<tr>";
        echo '<td colspan="2"  align="center">  3. zelle </td>';
        echo "</tr>";
              }
        echo "</table>";
   ?>
<?php   phpinfo(); ?>
```

http://mwilhelm.hs-harz.de/scripte/php/bsp05.php

HTML-Tabellen durch PHP

```
<html>
<body>
<table>
<?php
      for ($i=0;$i<5;$i++) {
      echo "<tr>";
      echo "<td>  $i 1. Zelle </td>";
      echo "<td>  2. Zelle </td>";
      echo "</tr>";
      echo "<tr>";
      echo '<td colspan="2"  align="center">  3. zelle </td>';
      echo "</tr>";
           }
      echo "</table>";
   ?>
<?php   phpinfo(); ?>
```

http://mwilhelm.hs-harz.de/scripte/php/bsp06.php

echo / echo

- Variablen werden in einfache Hochkommas nicht ersetzt
- Variablen werden in doppelte Hochkommas ersetzt

```
$name="meier";
echo  '$name';          // Ausgabe $name, ohne Zeilenumbruch, <br />
echo  "$name";          // Ausgabe meier
```

- PHP-Sonderzeichen müssen beachtet werden:

```
$name="meier";
echo "Der Name lautet " . $name;
echo "Der Name lautet " , $name;          // Fehler
echo "Der Name lautet " , $name;
```

echo / echo

XHTML zwingt zu Hochkommas im HTML-Code

```
    <input type="text" name="nachname" value="Anton" size="12" />
echo "<input type=\"text\" name=\"nachname\" value=\"Anton\" size=\"12\" />";
    // hier ist die Begrenzung das ' Zeichen
echo '<input type="text" name="nachname" value="Anton" size="12" />';
```

Lösung:

```
    $item="text";
    echo ' <input type="$item">';              // keine Ersetzung
    echo '< input type  =" ' . $item  . ' ">';  // Aufteilen der Strings
    echo '< input type  =" ' , $item , ' ">';   // Aufteilen der Strings mit Kommata
    echo "< input type =\"$ item\">";           // Sonderzeichen \
```

Typen, Variablen, Kontrollstrukturen, Operatoren

- PHP hat ein besonders einfaches Handling von Variablen
- dynamische Erzeugung durch Benutzung
  - Deklaration und Definition entfallen
  - guter Stil: dennoch deklarieren und definieren!
- schwach getypt, jederzeit ist ein Typwechsel erlaubt
  - guter Stil: Typen wahren, Wechsel vermeiden
- type casting: (typ)$variable und settype($variable, typ)
  - Typ ermitteln:
  - gettype($variable), is_int($variable), is_float($variable)..
- bei Bedarf: automatische Typwandlung
  - Vorsicht: im Extremfall eine versteckte Fehlerquelle

Typen, Variablen, Kontrollstrukturen, Operatoren

- define('KONSTANTE' , 'Mein Wert bleibt fix.');
- echo KONSTANTE; *// ohne $*
- Ausgabe: Mein Wert bleibt fix.

- <?php
- echo "PI als Konstante" ;
- define("PI", "3.141592653");
- define("lf", "
\r\n");
- echo (PI . lf);
- echo ("Datentyp von PI: " . gettype(PI));
- ?>

Einmal definierte Konstanten sind nicht mehr zu ändern.

http://mwilhelm.hs-harz.de/scripte/php/bspPI.php

Typen, Variablen, Kontrollstrukturen, Operatoren

Zusammengesetzte Variablen - Arrays

Eine Array-Variable enthält Speicherplatz für eine Liste von Werten.

Indizierte Arrays: Zugriff über Indexwert

- $a[0] = 7;
- $a[3] = "Hallo";
- $a[] = "erstmal";
- echo $a[4];
- Ausgabe: **???**

- Größe auslesen mit: echo count($a); Ergebnis: 3
- Mehrdimensionale Zuweisung z. B.: $b[0][0]=2;

Typen, Variablen, Kontrollstrukturen, Operatoren

Zusammengesetzte Variablen - Arrays

Eine Array-Variable enthält Speicherplatz für eine Liste von Werten.

Indizierte Arrays: Zugriff über Indexwert

- $a[0] = 7;
- $a[3] = "Hallo";
- $a[] = "erstmal";
- echo $a[4];
- Ausgabe: **erstmal**

- Größe auslesen mit: echo count($a); Ergebnis: 3
- Mehrdimensionale Zuweisung z. B.: $b[0][0]=2;

Typen, Variablen, Kontrollstrukturen, Operatoren

- Integer in PHP entsprechen dem Datentyp int in Java. Auf einer 32-Bit Maschine entspricht dies: -2.147.482.648 bis +2.147.482.647 ((-2^31-1) bis +2^31)
- Bei einem Überlauf wandelt PHP den Typ automatisch nach Double
- Es gibt eine dezimale (Basis 10), hexadezimale (Basis 16) und oktale (Basis 8) Notation für Integer. Bitte beachten, dass der Bereich hexadezimal und oktal notierter Integer nicht dem maximalen Wert eines dezimal notierten Integer Wertes entspricht
- // Integer-Range bei einer 32 Bit Maschine
  $min_integer = -2147483648;
  $max_integer = 2147483647;
- // Oktale Notation (Basis 8), Wert: 10 (dezimal)
  $octal = 012;
- // Hexadezimale Notation (Basis 16), Wert: 3605 (dezimal)
  $hex = 0xE15;

Typen, Variablen, Kontrollstrukturen, Operatoren

Weitere Typen

Zu den zusammengesetzten Typen zählen:

- Array (keine Grenze muss definiert werden)
- Object

Weitere Typen, die gettype() seit PHP 4 meldet sind:

- NULL
- Resource ID

Typen, Variablen, Kontrollstrukturen, Operatoren

Die Operatoren in PHP sind sehr ähnlich zu C, Perl oder Java

Ausdrücke:

. + - * / % & | ~

+= .= –= *= /= %= &= |=

< <= == === => > !=

$v++ ++$v $v-- --$v

Logische Verknüpfungen:

&& And

|| Or

Seit PHP 4 testet "===" auf Wert- und Typgleichheit (Identität). "==" in PHP 3 nur auf Wertgleichheit

Typen, Variablen, Kontrollstrukturen, Operatoren

Strings, Zeichenketten

– Stringoperatoren zum Verbinden von Strings

- $a="Hallo " . "Welt!";
- $a="Hallo "; $a .= "Welt";

Strings mit " " bzw. ' '

- $a= "Dies ist der Wert b $b";

  Ersetzung aller erkannten Variablen durch ihre Werte
- $a= 'Dies ist der Wert b $b';

  Keine Ersetzung der Variable, Behandlung als Strings

Arrays und Zusatzfunktionen

Beispiel

```
$mitarb = array("Mueller", "Schulze", "Meier", "Gates");
$anzahl=count($mitarb);
for ($i=0; $i<$anzahl; $i++) {
   echo ("Nr: $i:  $mitarb[$i] .lf );
}
sort($mitarb);
for ($i=0; $i<$anzahl; $i++) {
   echo ("Nr: $i:  $mitarb[$i] .lf );
}
```

Arrays und Zusatzfunktionen

Stringfunktionen

explode: Zerlegen eines Strings anhand Trennzeichen (Gegenstück implode)

- $staedte = "Stuttgart -> Esslingen ->Filderstadt";
- $orte= explode(" -> ", $staedte);

echo $orte[0]; Ausgabe Stuttgart

echo $orte[2]; Ausgabe Filderstadt

strcmp : Vergleichen von Strings

- $ergebnis=strcmp($a,$b);
- 0 bei gleich, -1 bei $a<$b, +1 bei $a>$b

Typen, Variablen, Kontrollstrukturen, Operatoren

Stringfunktionen

strlen: Bestimmen der Länge einer Zeichenkette

- echo strlen("Hallo"); Ausgabe : 5

str_replace: Ersetzen eines Strings

- $satz = 'Der Studiengang IB ist gut, aber…';

echo str_replace('IB','PSC', $satz);

Ausgabe: Der Studiengang PSC ist gut …

Viele weitere Beispiele (siehe z.B. php.net)

Typen, Variablen, Kontrollstrukturen, Operatoren

Stringfunktionen

- **echof** **formatierte Ausgabe %d, %u, %f, %s, %c**

Switch case – Anweisung **à la Java**

For-Schleife **à la Java**

For-each-Schleife **à la Java**

While-Schleife **à la Java**

Do-While-Schleife **à la Java, Bed. Am Ende**

Typen, Variablen, Kontrollstrukturen, Operatoren

if (then) else

```
if (Bedingung) {
    Anweisung(en)
}
else {
    Anweisung(en)
}

if (Bedingung)
    Anweisung1;
else
    Anweisung2;
```

Beispiel:

```
$zahl=5;
if($zahl>4) {
  echo "Größer als   4";
}
else {
  echo "Kleiner gleich 4";
}
```

Typen, Variablen, Kontrollstrukturen, Operatoren

Switch

```
switch(Variable){
    case Wert:
        Anweisung(en);
        break;
    …
    case Wert:
        Anweisungen;
        break;
    default:
        Anweisungen
}
```

Beispiel

```
$ort="Stuttgart";
switch($ort){
  case "Bottrop":
    echo "nicht schön!";
    break;
  case "Stuttgart":
    echo "sehr schön!";
    break;
  default:
    echo "naja!";
}
```

Typen, Variablen, Kontrollstrukturen, Operatoren

For - Schleife

```
for( Initialisierung(en); Bedingung; Anweisung(en))
   {
     Anweisung(en)
   }
```

Beispiel:

```
   for ($i=1; $i<100; $i++) {
     $k = $i * $i;
     echo "$i zum Quadrat ist $k <br />\n";
    }
```

break, continue

Typen, Variablen, Kontrollstrukturen, Operatoren

While - Schleife

Struktur:

```
while (Bedingung) {
  Anweisung(en)
}
```

Beispiel:

```
$i=1;
$k=1;
while($k<=100) {
  echo "$i zum Quadrat ist $k <br />";
  $i++;
  $k = $i * $i;
}
```

break, continue

Typen, Variablen, Kontrollstrukturen, Operatoren

Do - While - Schleife

```
do {
     Anweisung(en)
  }
 while (Bedingung);
```

Beispiel:

```
$i=$k=1;
do {
        $k=$i*$i;
        echo "$i zum Quadrat
        ist $k<br />";
        $i++;
} while ($k<100);
```

Unterschied zur while-Schleife:

Sie wird mindestens einmal durchlaufen.

break, continue

Typen, Variablen, Kontrollstrukturen, Operatoren

Foreach - Schleife

- foreach(Array-Variable as Variable){
- Anweisungen }
- In den Anweisungen jeweils Zugriff auf Variable möglich
- Zum Durchlaufen eines Arrays
- Beispiel:

```
$farbenliste=array('blau','rot','gelb');
foreach($farbenliste as $einzelfarbe){
  echo "Die Farbe ist: $einzelfarbe";
}
```

- Für assoziative Arrays (Ausgabe Schlüssel möglich):
- $farben = array('f1'=>'blau','f2'=>'rot','f3'=>'gelb');
- foreach($farben as $schluessel=>$farbe) {
- echo "Farbe mit Schlüssel $schluessel ist:
- $farbe
";}

2. Übung

- Erstellen Sie ein Programm,
- dass die Zahlen 1-50 ausgibt und jeweils
- dahinter angibt, ob die Zahl gerade ist oder
- nicht.

```
<html>
<head>
    <title></title>
    <meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
</head>
<body><p>
<?php
    for($i = 1; $i <= 50; $i++)
    {
        if ($i % 2 == 0)
            echo("$i ist gerade");
        else
            echo ("$i ist ungerade") ;
         echo("<br />\r\n");
    }
?>
</p></body>
</html>
```

http://mwilhelm.hs-harz.de/php/bsp50.php

Funktionen

- Funktionen können rekursiv aufgerufen werden
- Die Argumente werden normalerweise "per value" übergeben
- Sie können jedoch auch als Referenz übergeben werden, wenn dem Argumentnamen ein "&" vorangestellt wurde.
- Das Keyword "static" dient zur Deklaration von lokalen Funktionsvariablen, die zwischen Funktionsaufrufen ihren Wert behalten.

- Beispiel:

```
// Rekursion ist erlaubt
function fak($n) {
  if ($n>1)
    return $n*fak($n-1);
 else
   return 1;
}
```

Funktionen

- Funktionen - Beispiel

```
<html>
<body>
 <?php
    $a=2;
    $b=3;
    $c=addieren($a,$b);
    echo ( "Summe von $a und " . $b . " ist "  . $c );
    function addieren ($summand1, $summand2)
    {
        return $summand1 + $summand2;
    }
 ?>
</body>
</html>
```

http://mwilhelm.hs-harz.de/php/bspAdd1.php

Funktionen

- Funktionen – Beispiel: call by Reference

```
<html>
<body>
 <?php
    $a=2;
    $b=3;
    addieren($a,$b,&$c);
    echo ( "Summe von $a und " . $b  . ": "    .   $c );
    function addieren ($summand1, $summand2, &$erg)
    {
          $erg = $summand1 + $summand2;
    }
 ?>
</body>
</html>
```

http://mwilhelm.hs-harz.de/php/bspAdd2.php

Funktionen mit Referenzen:

```
function add ($a, $b, &$c) {
        $c=$a+$b;
}

$x=1;
$y=2;
$z=1;
add($x, $y, $z);
echo $z;
```

```
function add (&$c) {
        $c .= 'a²+b²=c²';
}

$sStr='Pythagoras: ';
add($sStr);
echo $sStr;
```

Funktionen

- Funktionen und Variablensichtbarkeit
  - <html><body>
  - <?php
  - $a=2;
  - $b=3;
  - echo "Hier die Summe: " . addieren();
  - function addieren()
  - {
  - $b = 7;
  - return $a + $b;
  - }
  - ?>
  - </body></html>

- Ergebnis: 9 ??

http://mwilhelm.hs-harz.de/scripte/php/bspAdd3.php

Funktionen

- Funktionen und Variablensichtbarkeit

```
<html><body>
<?php
  $a=2;
  $b=3;
  echo "Hier die Summe: " , addieren();
  echo "Hier der Wert von b:  $b" ;

  function addieren()
  {
     global $a;          // Nachteil der "Nichtdeklaration"
     $b = 7;
     return $a + $b;
  }
?>
</body></html>
```

- Ergebnis: 9

http://mwilhelm.hs-harz.de/php/bspAdd4.php

Funktionen:

Beispiel einer Rekursion

```
function zaehlen ($zahl) {
    if ($zahl == 100)
        echo 'fertig <br />';
    else
    {
        echo $zahl. '<br />';
        zaehlen($zahl + 1);
    }
    return $zahl;
}
zaehlen(1);    Ausgabe: 1-100, dann Abbruch, oder ?
```

Funktionen: Verbesserte Version

```
<?php
function zaehlen ($zahl) {
    if ($zahl < 100) {
        echo $zahl. '<br />';
        return zaehlen($zahl+1);
    }
    else
    {
        echo 'fertig <br />';
        return $zahl;
    }
}
echo zaehlen(1);
?>
```

addcslashes
addslashes
bin2hex
chop
chr
chunk_split
convert_cyr_string
count_chars
crc32
crypt
echo
explode
get_html_translation_table
htmlentities
htmlspecialchars
implode
join
levenshtein
ltrim
md5
metaphone
nl2br
ord
parse_str
echo
echof
quoted_echoable_decode
quotemeta
rtrim
setlocale
similar_text
soundex
sechof
sscanf
strcasecmp
strchr
strcmp
strcspn
stripcslashes
stripslashes
strip_tags
stristr
strlen
strnatcasecmp
strnatcmp
strncmp
strpos
strrchr
strrev
strrpos
strspn
strstr
strtok
strtolower
strtoupper
strtr
str_pad
str_repeat
str_replace
str_rot13
str_shuffle
str_split
str_word_count
substr
substr_count
substr_replace
trim
ucfirst
ucwords
wordwrap

Formulare in PHP

Formulare und Variablen

- Eingaben von Werten in HTML-Formularen, können an ein PHP-Skript gesendet werden.
- PHP-Skript kann diese dann auslesen und verarbeiten.

Beispiel-Formular:

```
<form action="VariablenFormular.php" method="post">
   Vorname: <input type="text" name="vorname"></input><br />
   Nachname: <input type="text? name="name"></input><br />
   Lieblingszahl: <input type="text" name="zahl"></input><br />
   <input type="submit"></input>
</form>
```

Formulare in PHP

Formulare auslesen

- $_POST['Variablenname'] zum Auslesen

  Zum Beispiel: $vorname=$_POST[‘vorname’];
- Variable entspricht Eingabefeldnamen im Formular
- WICHTIG: Eingaben prüfen (Sicherheit) ! “Never trust user input”
- isset, is_numeric, empty

Andere Übertragungsmethode:

- GET

  Im Formular: <form action="? " method="get">

  Auslesen: $_GET['Variablenname']
- Übertragung der Daten im Klartext in der URL

  http://localhost/VariablenFormular.php?passwort=Horst&login=...
- isset, is_numeric, empty

Beispielformular in PHP

```
if ( isset($_GET['vorname']) ){                    Ist gesetzt?
    $vorname = $_ GET['vorname'];
 }
 if ( isset($_ GET['nachname']) ){                 Ist gesetzt?
    $nachname = $_ GET['nachname'];
 }
 if ( isset($_ GET['zahl']) && is_numeric($_ GET ['zahl']) )
 {
    $zahl=$_ GET['zahl'];
 }
 else {
    $zahl="Keine Zahl";
 }
 echo "$vorname $nachname hat Lieblingszahl $zahl?;
```

Gibt es ein Problem?

Beispielformular in PHP

```
if ( isset($_POST['vorname']) ){                    Ist gesetzt?
    $vorname = $_POST['vorname'];
}
if ( isset($_POST['nachname']) ){                   Ist gesetzt?
    $nachname = $_POST['nachname'];
}
if ( isset($_POST['zahl']) && is_numeric($_POST['zahl']) )
{
    $zahl=$_POST['zahl'];
}
else {
    $zahl="Keine Zahl";
}
echo "$vorname $nachname hat Lieblingszahl $zahl?;
```

Gibt es ein Problem?

Abfrage eines Parameters in PHP

```
<?php
$VARS_COUNT = 0;
if ($REQUEST_METHOD == 'POST') {
  echo "post <br />";
  while (list($var, $value) = each($HTTP_POST_VARS))
  {
    echo "$var=$value <br />";
    $VARS_COUNT ++;
  }
}
else {
  echo "else <br />";
  while (list($var, $value) = each($HTTP_GET_VARS))
   {
    echo "$var=$value <br />";
    $VARS_COUNT ++;
  }
}
if ($VARS_COUNT == 0){
  echo "<p>No parameters transfered to
script.</p>";
}
else {
  echo "<p><b>$VARS_COUNT</b>
parameters was transfered to script by
<b>$REQUEST_METHOD</b>
method.</p>";
}

?>
```

http://mwilhelm.hs-harz.de/scripte/php/bsp05.php?value=1&w=abc

Abfrage eines Parameters in PHP

```
if ($VARS_COUNT == 0){
  echo "<p>No parameters transfered to
script.</p>";
}
else {
  echo "<p><b>$VARS_COUNT</b>
parameters was transfered to script by
<b>$REQUEST_METHOD</b>
method.</p>";
}

?>
```

Aufbau eines Formulars: test_form1.php

```
echo '<form method="get" action="http://www.a/test.php"> ';
echo "<p>eingabe:";
echo ' <input type="text" name="eingabe"> ';
echo "</p>";
echo "<br />";
echo "<p><input type=\"submit\" value=\" submit\"> ";
echo "<input type=\"reset\" value=\"loeschen\">";
echo "</p>";
echo "</form> ";
```

Aufbau eines Formulars: test_form2.php

```
<html>
<head>
  <title>2. Beispiel Formulare mit php</title>
</head>
<body>
<h2>2. Beispiel Formulare mit php</h2>
<?php
 echo "<form name=\"anmeldung\"  >\r\n";
 echo "geben sie ihren Benutzernamen an: <p>\r\n";
 echo "<input type=\"text\"   name=\"student\"   value=\"andreas\" size=30> \r\n";
 echo "<br /><br />\r\n";
 echo "Auswahl<br />\r\n\r\n";
</body>
</hmtl>
```

Aufbau eines Formulars: test_form2.php

```
echo "<select name=\"objekte\" >\r\n";
echo "<option value=\"Cb\">  Cube </option>\r\n";
echo "<option value=\"Keg\" selected=\"selected\" > Kegel\r\n";
echo "<option value=\"Zyl\" selected=\"selected\" > Zylinder\r\n";
echo "<option value=\"Qd\" > Quader\r\n";
echo "<option value=\"Kug\"> Kugel\r\n";
echo "<option value=\"Tor\"> Torus\r\n";
echo "</select>\r\n";
echo "<br /><br />\r\n";
echo "<input type=\"submit\" value=\"senden\"   >\r\n";
echo "<input type=\"reset\" value=\" delete \"><br />";
echo "</form> ";
?>
<br />                </body>              </html>
```

Aufbau eines Formulars: test_form3.php (submit)

```
echo "<form name=\"Anmeldung\" method=\"get\"
    action=\"http://mwilhelm.hs-harz.de/scripte/php/test_form3a.php\">\r\n";
 echo "Geben Sie Ihren Benutzernamen an: <p>\r\n";
 echo "<input type=\"text\"   name=\"student\"   value=\"andreas\" size=30> \r\n";
 echo "<br /><br />\r\n";
 echo "Auswahl<br>\r\n\r\n";
 echo "<select name=\"objekte\" >\r\n";
 echo "<option value=\"Cb\">  Cube </option>\r\n";
 echo "<option value=\"Keg\" selected=\"selected\" > Kegel\r\n";
 echo "<option value=\"Zyl\" selected=\"selected\" > Zylinder\r\n";
 echo "<option value=\"Qd\" > Quader\r\n";
 echo "<option value=\"Kug\"> Kugel\r\n";
 echo "<option value=\"Tor\"> Torus\r\n";
 echo "</select>\r\n";
```

Aufbau eines Formulars: test_form3a.php (submit)

```
$VARS_COUNT=0;
if (is_array($HTTP_GET_VARS) ) {
  echo "get-Method <br />";
  foreach ($HTTP_GET_VARS  as $index => $value) {
    $VARS_COUNT ++;
    echo "Index:  $index,   $VARS_COUNT  $value";
    echo "<br />";
  }
}
```

Dateioperationen in PHP: fopen

- Öffnen einer Datei
  - Read r
  - Write w
  - Read / Write r+
  - Write / Read w+
  - Read / Append a+ Cursor an das Ende der Datei
  - Append a

Beispiel:

- $ourFileName = "testFile.txt";
- $fh = fopen($ourFileName, 'X') or die("Can't open file");
- fclose($fh);

Dateioperationen in PHP: fread

- Lesen aus einer Datei
  - $var = fread(handle, AnzBytes)

Beispiel:

- $myFile = "test.txt";
- $fh = fopen($myFile, 'r');
- $theData = fread($fh, 5);
- fclose($fh);

echo $theData; **Hallo**

Dateioperationen in PHP: fread

- Komplettes Lesen aus einer Datei
  - $var = fread(handle, filesize(handle))

Beispiel:

- ```
  $myFile = "test.txt";
  $fh = fopen($myFile, 'r');
  $theData = fread($fh, filesize($myFile) );
  fclose($fh);

  echo $theData;
  ```
  ```

Dateioperationen in PHP: fread

- Zeilenweises Lesen aus einer Datei, sucht \n != \r\n
  - $var = fgets(handle)

Beispiel:

- ```
  $myFile = "test.txt";
  $fh = fopen($myFile, 'r');
  for ($i=1; $i<10; $i++) {
        $theData = fgets($fh);
        echo $theData;
  }
  fclose($fh);
  ```
  ```

Dateioperationen in PHP: fwrite

- Schreiben in eine Datei
  - fwrite(handle, Variable)

Beispiel:

- $myFile = "test.txt";
- $fh = fopen($myFile, 'w') or die("can't open file");
- $strData = "Bald ist Semesterende\n";
- fwrite($fh, $strData);
- $strData = "und es gibt Klausuren\n";
- fwrite($fh, $strData);
- fclose($fh);

Beispiel: bsplistfile.php
Datei: list.dat
Aufgabe: Darstellen als Tabelle

- **Christopher;chris@freenet.com;3065 Cumberland Circle**
- **Joseph;joseph@h1.net;501 Glen Road**
- **Marco;marco@mail.org;305 Adam**
- **Fernand;fernand@hmail.net;81 Mechanic Street**
- **Stephan;stephan@itservice.de;Friedhofsallee 93**

Dateioperationen in PHP: Lesen und Parsen

```
<?php
 $filename = "list.dat";
 $lines = file($filename);  // Alle Zeilen eingelesen
 for ($i=0;$i<count($lines);$i++){
   list($name, $email, $adress) = split(";",$lines[$i]);
   $name = trim($name);
   $email = trim($email);
   $adress = trim($adress);
   echo "<tr>\n";
   echo "  <td>$name</td>\r\n";
   echo "  <td><a href=\"mailto:$email\">$email</a></td>\r\n";
   echo "  <td>$adress</td>\r\n";
   echo "</tr>\r\n\r\n";
 }
?>
```

Dateioperationen in PHP: Lesen und Parsen

```
<?php
 $filename = "list.dat";
 $lines = file("$filename");              // Alle Zeilen eingelesen
 for ($i=0;$i<count($lines);$i++){
   $adr= explode(";", $lines[$i] );    // Trennen nach ;
   $name = trim($adr[0]);
   $email = trim($adr[1]);
   $adress = trim($adr[2]);
   echo "<tr>\n";
   echo "  <td>$name</td>\r\n";
   echo "  <td><a href=\"mailto:$email\">$email</a></td>\r\n";
   echo "  <td>$adress</td>\r\n";
   echo "</tr>\r\n\r\n";
 }
?>
```

Dateioperationen in PHP: bsp_listfile.php

```
<table border="1">
<tr>
  <td><b>Name</b></td>
  <td><b>Email</b></td>
  <td><b>Address</b></td>
</tr>

<?php
  $filename = "list.dat";
  $lines = file("$filename");
  for ($i=0;$i<count($lines);$i++){
    list($name, $email, $adress) = split(";",$lines[$i]);
    $name = trim($name);
    $email = trim($email);
    $adress = trim($adress);
    echo "<tr>\n";
    echo "  <td>$name</td>\n";
    echo "  <td><a href=\"mailto:$email\">$email</a></td>\n";
    echo "  <td>$adress</td>\n";
    echo "</tr>\n\n";
  }
?>
</table>
```

Dateioperationen in PHP: bsp_listfile.php

Name	Email	Address
Christopher	chris@freenet.com	3065 Cumberland Circle
Joseph	joseph@h1.net	501 Glen Road
Marco	marco@mail.org	305 Adam
Fernand	fernand@hmail.net	81 Mechanic Street
Stephan	stephan@itservice.de	Friedhofsallee 93

Datum in PHP

```
<?php
  echo "<p>Current Date: <br /><br /><strong>";
  echo date("D d.m.Y G:i:s") . "<br />";
  echo date("F j, Y, g:i a") . "<br />";
  echo date("m.d.y") . "<br />";
  echo date("j, n, Y") . "<br />";
  echo date("Ymd") . "<br />";
  echo date('h-i-s, j-m-y, it is w Day z ') . "<br />";
  echo date('\i\t \i\s \t\h\e jS \d\a\y.') . "<br />";
  echo date("D M j G:i:s T Y") . "<br />";
  echo date('H:m:s \m \i\s\ \m\o\n\t\h') . "<br />";
  echo date("H:i:s") . "<br />";
  echo "</strong></p>\n";
?>
```

Session-Variablen

- **Alte Technik**
  - Hidden-Elemente
  - Weitergabe über den Parameter
- **Neue Technik**
  - Session-Variablen mittels PHP
  - Dem Aufrufer wird eine eindeutige **Session-ID** zugeordnet. Somit kann man in PHP den Aufrufer genau identifizieren. Die Session ID wird entweder als Cookie gespeichert oder an die URL gehängt (Parameter).
- Man beliebig viele Variablen definieren

Funktionen bezüglich Session-Variablen

- on_cache_expire — Liefert die aktuelle Cache-Verfallszeit
- session_cache_limiter — Liefert und/oder setzt die aktuelle Cacheverwaltung
- session_commit — Alias von session_write_close
- session_decode — Dekodiert die Daten einer Session aus einer Zeichenkette
- session_destroy — Löscht alle in einer Session registrierten Daten
- session_encode — Kodiert die Daten der aktuellen Session als Zeichenkette
- session_get_cookie_params — Liefert die Session-Cookie Parameter
- session_id — Liefert und/oder setzt die aktuelle Session-ID
- session_is_registered — Überprüft, ob eine globale Variable in einer Session registriert ist
- session_module_name — Liefert und/oder setzt das aktuelle Session-Modul
- session_name — Liefert und/oder setzt den Namen der aktuellen Session

Funktionen bezüglich Session-Variablen

session_regenerate_id	Ersetzt die aktuelle Session-ID durch eine neu erzeugte
session_register_shutdown	Funktion zum Schließen von Sitzungen
session_register	Registriert eine oder mehrere globale Variablen in der aktuellen Session
session_save_path	Liefert und/oder setzt den aktuellen Speicherpfad der Session
session_set_cookie_params	Setzt die Session-Cookie-Parameter
session_set_save_handler	Setzt benutzerdefinierte Session-Speicherfunktionen
session_start	Initialisiert eine Session
session_status	Gibt den Status der aktuellen Sitzung zurück
session_unregister	Hebt die Registrierung einer globalen Variablen in der aktuellen Session auf
session_unset	Löscht alle Session-Variablen
session_write_close	Speichert die Session-Daten und beendet die Session

Session-Variablen:

- **Start**
  - 1. Zeile der Datei
  - <?php session_start(); ?>
- **Test, ob die Variable existiert**
  - if (!isset($_SESSION["var1"])) $_SESSION["var1"]=0;
- **Wert abfragen**
  - $var1 = $_SESSION["var1"];
- **Wert erhöhen**
  - $var1 = $_SESSION["var1"];
  - $var1++;
  - $_SESSION["var1"] = $var1;
- **Wert erhöhen**
  - $_SESSION['var1']++;
- **Wert ausgeben**
  - echo $_SESSION["var1"];
- **Session löschen**
  - unset ($_SESSION["var1"]);

Session-Variablen: 1. Beispiel

Session-Variablen: 1. Beispiel

Session-Variablen: 1. Beispiel

Session-Variablen: 1. Beispiel: HTML

```
<title> Startformular für eine Session-Beispiel </title>

<!-- http://mwilhelm.hs-harz.de/scripte/php/session_formular.html  -->
<h3>  Startformular für eine Session-Beispiel</h3>

<form action="session_1.php" method="post">
  Name: <input type="Text" name="name"  value="unbekannt" size="12"><br />
  <br />
  <input type="submit" value="senden" />
  <input type="reset" value="Löschen"/><br />

</form>
```

Session-Variablen: 1. Beispiel: 1. PHP-Seite

```
<?php
session_start(); //Ganz wichtig
?>
<?php
// bestimmen, ob schon registriert
$name = $_POST['name'];
if(!isset($name)) {
   $name = "Gast";
}
//Sessionsvariable registieren
  $_SESSION['username'] = $name;
    //Text ausgeben
  echo "Sie heißen $name <br>\n";
  echo "<a href=\"session_2.php\">Weiter zur 2. Seite</a>\n";
?>
```

Session-Variablen: 1. Beispiel: 2. PHP-Seite

```
<?php
session_start(); //Ganz wichtig
?>


<?php
    //In $name den Wert der Sessionvariablen speichern
    $name = $_SESSION['username'];


   //Text ausgeben
  echo "Sie heißen immer noch: $name\n";
?>
```

Sie heißen immer noch: <?php $name\n"; ?>

Session-Variablen: 2. Beispiel: 2. PHP-Seite

```
<?php
session_start(); //Ganz wichtig
?>
<?php
// bestimmen, ob schon registriert ?
$name = $_POST['name'];
if(!isset($name)) {
   $name = "Gast";
}
//Sessionsvariable registieren
$_SESSION['username'] = $name;
//Text ausgeben
echo "Sie heißen $name <br>\n";
echo "<a href=\"session_2.php\">Weiter zur 2. Seite</a>\n";
?>
```

Session-Variablen: 2. Beispiel: 2. PHP-Seite

```
<?php
session_start(); //Ganz wichtig
?>

<html>
<body>

  <?php
    //In $name den Wert der Sessionvariablen speichern
    $name = $_SESSION['username'];

    //Text ausgeben
   echo "Sie heißen immer noch: $name\n";
 ?>
</body>
</html>
```

Session-Beispiele

- **1. Beispiel**
  - session_formular.html
  - session_Aufgabe_1.php
  - session_Aufgabe_2.php
  - Aufruf:
    - http://mwilhelm.hs-harz.de/scripte/php/session_formular.html

- **2. Beispiel**
  - session_1.php
  - session_2.php
  - Aufruf:
    - http://mwilhelm.hs-harz.de/scripte/php/session_1.php

Literatur

Sebastian Bergmann:
Professionelle Softwareentwicklung mit PHP 5,
ISBN 3-89864-229-1

Markus Nix, Sandro Groganz:
Exploring PHP,
ISBN 978-3-935042-95-6

Jörg Krause:
Programmieren lernen in PHP4,
ISBN: 3-446-21754-1

Pollakowski, Martin:
Grundkurs MySQL und PHP,
ISBN 978-3-528-15829-3

Ferner, Jens:
Profikurs PHP5,
ISBN 978-3-8348-0155-5

Literatur

Avci, Trittmann, Mellis;
Web-Programmierung; Vieweg Verlag,
ISBN 3-528-05857-9

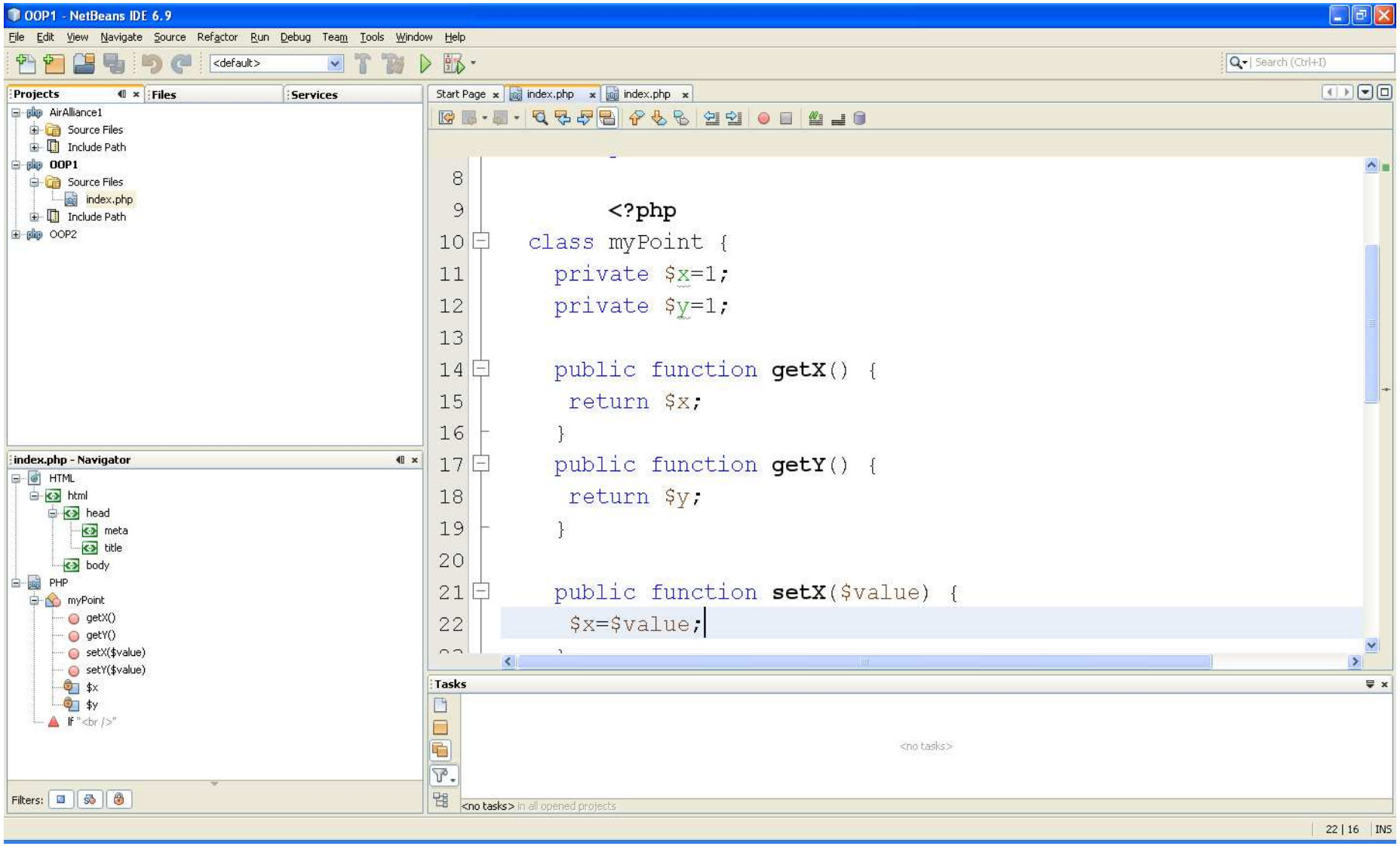
Buchmann, Smolarek:
PHP - interaktiv;
ISBN 3-936121-01-X

Kevin Yank:
PHP und MySQL;
ISBN 3-89864-198-8

Entwicklungsumgebungen

- PHP mit Eclipse
- **Netbeans IDE for PHP**
  - IDE mit Syntax-Highlight
  - Syntax Überprüfung
  - Automatischen Transfer zum Server
- ZEND Studio, www.Zend.com
- Maguma Studio von http://www.maguma.com
- Delphi for PHP

Netbeans


OOP1 - NetBeans IDE 6.9
File Edit View Navigate Source Refactor Run Debug Team Tools Window Help
Projects Files Services
AirAlliance1
Source Files
Include Path
OOP1
Source Files
index.php
Include Path
OOP2
index.php - Navigator
HTML
html
head
meta
title
body
PHP
myPoint
getX()
getY()
setX($value)
setY($value)
$x
$y
If "<br />"
Filters:
Start Page
index.php
index.php
8
9 <?php
10 class myPoint {
11 private $x=1;
12 private $y=1;
13
14 public function getX() {
15 return $x;
16 }
17 public function getY() {
18 return $y;
19 }
20
21 public function setX($value) {
22 $x=$value;
Tasks
<no tasks>
<no tasks> in all opened projects
22 | 16 INS