

Übung zur Vorlesung  
**Multimedia im Netz**

Doris Hausen  
Ludwig-Maximilians-Universität München  
Wintersemester 2009/2010

# Organisatorisches

- Informationen zu Vorlesung & Übung  
<http://www.medien.ifi.lmu.de/lehre/ws0910/mmn/>
- Übungsbetrieb
  - Anmeldung über UniWorx
  - Übungsblätter im eineinhalb Wochen Rhythmus (→ Drei normale Übungen, eine Tutorübung für Programmierfragen)
  - Regelmäßige Abgabe von Übungsblättern in Einzelarbeit (!!)  
(später auch Gruppenarbeit)
  - Bewertungsschema: „Nicht bestanden“ - „Bestanden“ - „Sehr Gut“
- Schein & Benotung
  - Klausur am Ende des Semester
  - Punkte aus den Übungsblättern verbessern die Note



# Einführung in PHP



- Serverseitige Skriptsprache seit 1995
- Aktuell in Version 5.3
- Offizielle Webseite: <http://www.php.net>
- Tutorials (sehr kleine Auswahl):
  - <http://tut.php-quake.net/de/index.html>
  - <http://www.php-einfach.de/php-tutorial/php-tutorial.php>

- **Beispiel:**

```
<html>
<head></head>
<body>
Dies ist die erste Zeile. <br>
Und dies ist die zweite Zeile. <br>
<?php
    echo "Und dies ist die dritte Zeile.<br>";
    $datum_uhrzeit=date("d.m.Y, H:i:s", time());
    echo "Heute ist der " . $datum_uhrzeit . " Uhr";
?>
<br>
Weiter geht's.
</body>
</html>
```

# Typen

- Schwache Typisierung
  - Boolean
  - Integer
  - Float
  - String
  - Array
  - Object
  - Resource
  - Null
- Variablen müssen nicht explizit deklariert werden

# Operatoren (Auswahl) (1)

- **Arithmetische Operatoren**  
+ (Addition); - (Subtraktion); \* (Multiplikation); / (Division); % (Modulus)
- **Bit-Operatoren**  
& (Und); | (Oder); ^(XOR); ~ (Nicht); << (Nach links verschieben); >> (Nach rechts verschieben)
- **Vergleichs-Operatoren**  
== (Gleich); === (Identisch, also gleich und vom selben Typ); != oder <> (Ungleich); !== (nicht identisch); < (kleiner als); > (größer als); <= (kleiner gleich); >= (größer gleich)

# Operatoren (Auswahl) (2)

- Inkrement- bzw. Dekrementoperatoren  
++\$a (Prä-Inkrement); \$a++ (Post-Inkrement); --\$a (Prä-Dekrement); \$a-- (Post-Dekrement)
- Logische Operatoren  
and alternativ && (Und); or alternativ || (Oder); xor (entweder oder); ! (nicht);
- Trinitätsoperator (vgl. If... then... else)  
(ausdr1) ? (ausdr2) : (ausdr3)  
Wenn “ausdr1” true ist, dann “ausdr2”; wenn “ausdr1” false ist, dann “ausdr3”

# Syntax

- PHP kann in HTML Dokumente eingebettet werden  
`<?php ... ?>`
- Variablen werden mit „\$“ deklariert und sind case-sensitive  
`$var = 5;`
- Textausgabe mit dem Befehl „echo“  
`echo "Text der ausgegeben werden soll";`  
`echo "Auch <b>HTML</b> kann so ausgegeben werden";`
- Konkatenation  
`echo "Inhalt der Variablen: " . $var;`
- Kommentare  
`// Das ist ein Kommentar`  
`/* Das ist auch ein Kommentar`  
`mit mehreren Zeilen sogar! */`

# Kontrollstrukturen (1)

- **if-else**

```
if ($a > $b) {  
    echo "a ist größer als b";  
} else {  
    echo "a ist NICHT größer als b";  
}
```

- **while**

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

- **do...while**

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

# Kontrollstrukturen (2)

- **for**

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

- **foreach**

```
foreach ($arr as $value) {  
    echo "Wert:  $value<br />\n";  
}
```

- **switch**

```
$i = 0;  
while (++$i) {  
    switch ($i) {  
        case 5:  
            echo "Fünf";  
            break 1; /* Beendet nur switch. */  
        case 10:  
            echo "Zehn";  
            break 2; /* Beendet switch und while. */  
        default:  
            break;}}
```

# Kontrollstrukturen (3)

- **break**

```
$i = 0;
while (++$i) {
    switch ($i) {
        case 5:
            echo "Fünf";
            break 1; /* Beendet nur switch. */
        case 10:
            echo "Zehn";
            break 2; /* Beendet switch und while. */
        default:
            break;}}}
```

- **continue**

```
while (list ($key, $value) = each ($arr)) {
    if (!(($key % 2)) { // überspringe ungerade Werte
        continue;
    }
    tue_was_mit_ungerade ($value);
}
```

# Funktionen

- PHP Funktion

```
function rechne_array($eingabe)
{
    echo "$eingabe[0] + $eingabe[1] = ", $eingabe[0]+$eingabe[1];
}
```

- Funktion mit Rückgabewert

```
function quadrat ($zahl)
{
    return $zahl * $zahl;
}

echo quadrat (4);    // gibt '16' aus.
```

- Klassen sind auch möglich

...aber nicht Teil dieser Übung ;)

# POST & GET

- Inhalte aus einem Formular können an PHP Skript übergeben werden

```
<form action="formular.php" method="post">  
    Name: <input type="text" name="username" /><br />  
    Email: <input type="text" name="email" /><br />  
    <input type="submit" name="submit" value="Und ab!" />  
</form>
```

- Zwei Methoden
  - Post: `$_POST[ 'username' ]`
  - Get: `$_GET[ 'username' ]`; Wert der Variable wird in URL angezeigt

# Sessions (1)

- Warum: Variablen-Inhalt ist bei Neuladen der Seite verloren
- Über Session-ID (gespeichert in Cookie oder URL) werden die Daten auf dem Server gespeichert und von dort geholt

- Session starten:

```
<?php session_start(); ?>
```

Wichtig: Muss ganz am Anfang stehen, vor jeglichem anderen HTML/PHP Code

- Variablen-Inhalte in Sessiondaten speichern

```
$_SESSION[ 'versuche' ] = 10;
```

# Sessions (2) - Beispiel

- Variable wird deklariert

```
<?php
    session_start();
    $_SESSION['zufallszahl'] = rand(0,100);
?>
<html>
    <head></head>
    <body>
        <h1>Das ist ein Beispiel für Sessions</h1>
        <p>
            <?php
                echo "Die vom Computer generierte Zufallszahl ist: " .
                    $_SESSION['zufallszahl'];
            ?>
        </p>
    </body>
</html>
```

# Sessions (3) - Beispiel

- Variable kann auf anderer Seite wieder ausgelesen werden

```
<?php
    session_start();
?>
<html>
    <head></head>
    <body>
        <h1>Das ist ein Beispiel für Sessions</h1>
        <p>
            <?php
                echo "Die vom Computer generierte Zufallszahl ist <b>immer
                noch</b>: " . $_SESSION['zufallszahl'];
            ?>
        </p>
    </body>
</html>
```

# PHP im CIP

- Version 5.2.6
- Skripte müssen unter public\_html/php abgelegt sein
- Skripte können aufgerufen werden über:  
<http://php.cip.ifi.lmu.de/~loginkennung/php/skriptname.php>
- Nur innerhalb des LRZ-Netz verfügbar!

# PHP am eigenen PC

- Installation eines Apache-Webserver nötig
- Beispielsweise: XAMPP
  - Distribution von Apache, MySQL, PHP und Perl
  - Verfügbar für Windows, Linux, Solaris und Mac)
  - <http://www.apachefriends.org/de/xampp.html>
  - Es empfehlen sich regelmäßige Updates von XAMPP um Sicherheitslücken vorzubeugen!

