TU Clausthal

01101
11010
10111
BIT
Business Information Technology
©TU Clausthal

Vorlesung
Werkzeuge der Informatik

Grundlagen und Werkzeuge des WWW (Teil 3)

Jörg P. Müller

Inhalt

- Entwicklung von Internet und WWW
- WWW-Architektur und Protokolle
  - Web Ressourcen (oder: Was ist eine URL)
  - Basisprotokoll des Internet: TCP/IP
  - WWW-Architektur (Client-Server)
  - Das HTTP-Protokoll
- Darstellung von WWW-Inhalten
  - Das WWW-Dokumentenmodell: HTML
  - Cascading Style Sheets (CSS)
- **Dynamische Erzeugung von Webseiten**
  - **Prinzipien**
  - **Die Skriptsprache PHP**

Dynamische Webinhalte

- **Bisher haben wir HTML-Seiten als statisch (d.h. vorab erzeugt) kennen gelernt**
- **Für viele Webanwendungen kann die Antwort auf eine Anfrage aber erst zum Zeitpunkt der Anfrage ermittelt werden**
  - **Google-Suchanfrage:**
    - **Menge der indizierten Webinhalte ändert sich laufend**
  - **Amazon-Produktsuche:**
    - **Momentane Inhalte des Produktkatalogs**
    - **Personalisierung bzgl. aktuellem Nutzer**
- **Wie die Beispiele zeigen, gilt dies insbesondere für alle datenbankbasierten Anwendungen**

Skriptsprachen

- **Werden hauptsächlich dazu verwendet, die Inhalte von Webseiten interaktiv (dynamisch) zu gestalten**
- **d.h. Elemente der Website werden während der Anzeige dynamisch verändert.**
- **Diese Veränderung kann duch das Einwirken des Nutzers oder automatisch geschehen.**
- **Die gebräuchlichsten Sciptsprachen sind JavaScript, JScript, VBScript, PHP, CGI und Perl.**
- **HTML-Dateien bei denen einzelne Elemente während der Anzeige dynamisch ihren Inhalt ändern, werden auch als *dynamisches HTML* bezeichnet.**

Eigenschaften von Skriptsprachen

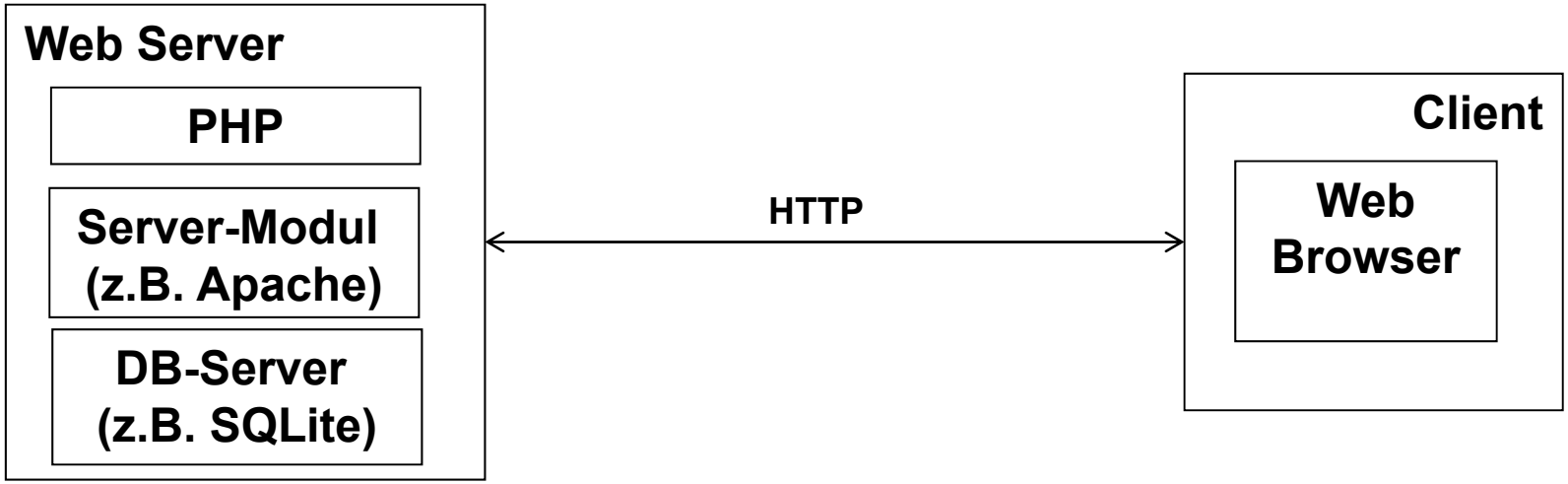
- Ausführung durch Interpreter
- Typfreiheit bzw. schwache Typisierung
- Oft weniger komplexe Datenstrukturen
- Oft Einbettung von Betriebssystemkommandos
- Komfortable Operationen mit Strings
- Einbinden externer bzw. fremder Programme, z.B.
  - Dateioperationen
  - Versenden von Emails
  - Datenbankanfragen

Serverseitige vs. clientseitige Skriptsprachen

- **Clientseitig:**
  - **Das Skriptprogramm ist in das vom Server gelieferte HTML-Dokument eingebettet**
  - **Die Ausführung erfolgt im Webbrowser**
  - **Beispiel: Javascript (siehe Teil 2), Visual Basic Skript**
  - **Gut geeignet z.B. zur lokalen Validierung von Nutzereingaben**
- **Serverseitig:**
  - **Die Erstellung des vom Server an den Client gesendeten HTML-Dokumentes erfolgt auf der Serverseite dynamisch**
  - **Das Skriptprogramm liegt auf dem Server; Der Client sieht nur das resultierende HTML-Dokument**
  - **Beispiel: PHP, Perl, Java Server Pages, beliebige ausführbare Programme (z.B. Shell-Skripte, Java Servlets)**
  - **Gut geeignet für die Erstellung von Seiten auf der Basis von Information aus Datenbanken**

Die Skriptsprache PHP

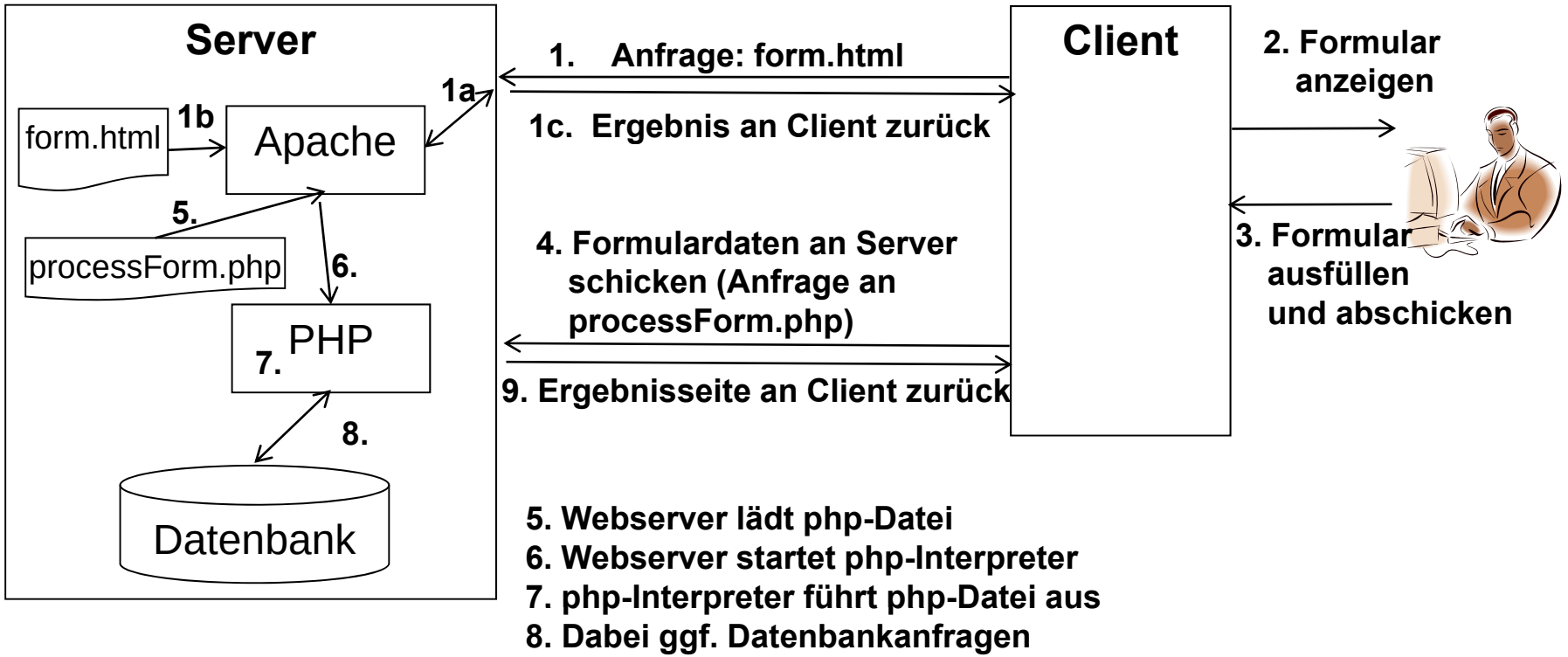
- **Steht für: PHP Hypertext Preprocessing**
- **Modul bzw. Programm, das auf den Webserver installiert ist**
- **Präprozessor: Verarbeitet PHP-Befehle (sogenannte PHP-Skripte) in einer Datei, bevor die Datei vom Web Server an den Client geschickt wird**



- **PHP-Skripte erstellt man am besten mit einem Text-Editor (z.B. Emacs, vi, Ultraedit, PHPEdit, Bluefish)**

Verwendung von PHP-Skripten

- **Typischerweise: Verarbeitung von Formulardaten**
- **Ablauf der Kommunikation zwischen Client und Server:**



Ein erstes PHP-Skript

```
<html>
<body>
<?php
/*
 Auch mein erstes php-Skript
 ist schon kommentiert
*/
echo 'Das kommt vom Server<hr>';
// hier endet Skriptcode
?>
</body>
</html>
```

- Das Sprachkonstrukt `echo` wird benutzt, um einen Text auszugeben

- PHP-Skript besteht aus
  - Reiner Text (z.B. HTML)
  - Skriptbegrenzer
  - PHP Skriptcode (Befehle)
- Im nebenstehenden Beispiel kommt folgender Code beim Client an:

```
<html>
<body>
Das kommt vom Server<hr>
</body>
</html>
```

Text- und HTML-Ausgabe mit PHP

- Sie können mit echo HTML-Steuerzeichen, wie z.B. '>' oder '<' und somit auch ganze HTML-Tags ausgeben.
- Versuchen Sie:

```
<?php
 echo 'Ich will eine Zeile
  ausgeben    ';
 echo 'Ich will noch eine
  Zeile ausgeben'; ?>
```

- Lösung:
  - HTML-Tag `<br/>` einfügen
- Wie gebe ich ein Steuerzeichen in php ein? z.B.

```
echo 'Er sagte 'Guten Tag' und
  ging';
```

- Lösung: Escape-Sequenz mit dem Zeichen '\'

```
echo 'Er sagte \'Guten Tag\'
  und ging';
```

- Sie können auch das Zeichen `'\'` escapen

Double-quoted vs. single-quoted strings

- Single-quoted strings verwenden den einfachen Apostroph
- `echo 'Er sagte \'Guten Tag\' und ging';`
- Double-quoted strings verwenden Gänsefüßchen
- `echo "Er sagte \'Guten Tag\' und ging";`
- In Double-quoted Strings sind zusätzliche Steuerzeichen möglich, z.B.
  - `\n` für Zeilenumbruch
  - \t für Tabulatur (Einrückung)
- Beispiel

```
<?php
  echo 'Er sagte \'Guten Tag\' \n und ging';
?>
```

Funktionen

- Aus einem PHP-Skript können Teilprogramme aufgerufen werden, die einen Wert zurückgeben

→ Funktionen

- Beispiel:
  - Die Funktion phpinfo() schreibt die aktuelle PHP-Konfiguration.
  - Ausprobieren!

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

- Allgemeine Form von Funktionen

```
<?php
  name_der_funktion(parameter1,
  parameter2,...);
?>
```

- Funktionen selber definieren

```
function hello($name) {
 $result='Hello, '.$name.'!';
 return hello;
}
```

Variablen und Verkettungsoperator

- **Variable:**
  - **Platzhalter, die im php-Skript dynamisch mit Daten gefüllt werden können**
  - **Beginnen mit '$', gefolgt von einem Buchstaben oder '_', danach beliebige Buchstaben, Zahlen, '_'**
  - **Groß- u Kleinschreibung zählt**
- **Mit dem Zuweisungsoperator '=' kann einer Variable ein Wert zugewiesen werden**
- **Mit dem Verkettungsoperator '.' kann man Strings und Variablen verketten**

```
<?php
 $vorname1='max';
 $nachname1='mustermann';
 $name1=$vorname1.' '.$nachname1;
 $zaehler=1;
 $vorname2='berta';
 $nachname2='buerger';
 $name2=$vorname2.' '.$nachname2;
 $zaehler=$zaehler+1;

echo 'Es gibt '.$zaehler.' Namen:
   '. $name1.' und '.$name2;
?>
```

Zahlen und Rechnen

- PHP unterstützt die Verwendung von Zahlen
- Dabei erkennt PHP zwei Typen von Zahlen
  - Ganze Zahlen (Integer)
  - Fließkommazahlen(Float)
- PHP unterstützt
  - Sechs Grundrechenoperationen
  - Weitere mathematische Funktionen, z.B.

```
sqrt(25); // 5
pow(2, 3); // 8
```

```
<?php
  $a = 10;
  $b = 6;
// die vier Grundrechenarten
  echo $a-$b;
  echo $a+$b;
  echo $a*$b;
  echo $a/$b;
// Komplement
  echo -$a;
// Modulo-Operation
  echo $a%$b;
?>
```

Kontrollflussanweisungen

- Datentyp Boolean → Werte `true` und `false`
- Verwendet zur Steuerung des Kontrollflusses in PHP-Skripten
- IF-Anweisung

```
<?php
  if (ausdruck) {
      anweisung_1;
      anweisung_2;
      // ...
      anweisung_n;
  }
  ?>
```

- `ausdruck` kann zusammengesetzter Ausdruck sein, mit den logischen Verknüpfungen `and`, `or`, `!` und `xor` verbunden
- IF-ELSIF-ELSE

```
<?php
  status=pruefe();
  if (status==1){
     // führe aktion1 durch
  } elsif (status==2){
     // führe Aktion 3 durch
  } else
    // zeige Fehlermeldung
  }
  ?>
```

Weitere Kontrollflussanweisungen

- **While-Schleife**

```
<?php
while (isAutoDreckig(
{
reinigen();
}
?>
```

- **Do-While-Schleife**

```
<?php
  do {
   verbinden();
   $daten = datenAbfragen();
   verbindungBeenden();
  } while ($daten > 0);
?>
```

- **For-Schleife**

```
<?php
  /* gib die Zahlen von
     0 bis 9 aus (bei 10
     ist die Bedingung false)
  */

  for ($i=0; $i<10; $i++) {
      echo $i."\n";
  }
?>
```

Arrays

- Verarbeitung von Listen und Matrizen

```
$farben=array('rot','gruen','blau');
```

- Arrayelemente können sein
  - Zahlen, Strings, Arrays

  → Mehrdimensionaly Arrays

```
$matrix=array(array(1,2),array(1,2));
```

- Zugriff auf Arrayinhalte über Index

```
echo $farben[0];
for ($i=0;i<sizeof($farben);i++){
    echo farben($i).' '; }
```

- Löschen eines Array-Elements

```
unset($farben[2]);
```

- Testausgabe von Arrays:

```
var_dump($farben);
```

- Assoziative Arrays
  - Index ist Stringwert

```
$nutzer = array(
  "Name"  => "Karl",
  "Alter" => 44,
  "Beruf"   => "Student"
);
echo "Ich bin
  ".$nutzer["Name"];
echo " und ".$nutzer["Beruf"];
```

Ein Beispiel: Funktionen und Arrays

```
<?php
function schreibeZeile($zeile){
  echo '<tr>';
  for ($i=0; $i<sizeof($zeile); $i++){
  echo '<td>'.$zeile[$i].'</td>';
  }
  echo '</tr>';
}
$kundendaten=array(array('kundenid','nachname','vorname', 'alter',
  'saldo'),
                   array('4711','meier','heinz',32, 10000),
                   array('0815', 'schulze', 'berta', 41, 50000),
                   array('1234', 'schmidt', 'karl', 38, 5000)
                  );
echo "<html><body>";
echo "<h1>Kundendaten</h1>";
echo "<table border=\"1\">";
for ($j=0; $j<sizeof($kundendaten); $j++){
  schreibeZeile($kundendaten[$j]);}
echo "</table></body></html>";
?>
```

Verarbeiten von Nutzereingaben

- PHP erlaubt die einfache Übermittlung von Nutzereingabedaten mit den HTTP-Methoden GET und POST
- Hierzu stehen zwei globale Array-Variable $_GET und $_POST zur Verfügung
- Bei GET werden Parameter an die URL gehängt:

```
http://www.mysite.com/
  registerUser.php?user=joerg&age=44
                   &married=true
```

- Zugriff auf die Werte im Skript registerUser.php:

```
echo $_GET['user'];
echo $_GET['age'];
echo $_GET['married']
```

- Alle Werte in $GET sind Strings oder Arrays

```
var_dump($_GET['age']);
// Ausgabe: String "44"
var_dump($_GET['married']);
// Ausgabe: String "false"
```

- Testen, ob bestimmte Werte gesetzt sind

```
if (! isset($_GET['user'])
  echo 'user not specified';
```

Beispiel: Verarbeitung von Formularen
Eingabeformular in HTML

```
<html>
 <HEAD><TITLE>Nutzer registrieren</TITLE></HEAD>
 <BODY>
  <form method="post" action="http://winf.in.tu-
  clausthal.de/jpm/php/registeruser.php">
   <fieldset>
    <legend>Registrieren Sie sich</legend>
    <label>Nutzername:
     <input name="user" type="text" size="10"></label>
    <label>Alter:
     <input name="age" type="text" size="3"  maxlength="3"></label>
    <label>Verheiratet:
     <fieldset>
      <label><input type="radio" name="married" value="true"
       />Ja</label>
      <label><input type="radio" name="married" value="false"/>
       Nein</label>
     </fieldset>
    </label>
    <input type="submit" value="Absenden">
    <input type="reset" value="Abbrechen">
   </fieldset>
  </form>
 </body>
</html>
```

Beispiel: Verarbeitung von Formularen
PHP-Skript zur Formularbearbeitung

```
<HTML>
<BODY>
<H1> Ihre Registrierung </H1>
 <?php
 // pruefe Parameter
 if (!isset($_POST['user'], $_POST['age'])) {
   die('Benutzen sie nur Formulare von der Homepage.');}
 $user=$_POST['user'];
 $age=$_POST['age'];
 if ($_POST['married']=="true"){
  $married="Ja";
 }else{
  $married="Kann ja noch werden";}
 echo "<h3> Herzlich Willkommen!</h3>";
 echo "<b>Ihr Nutzername:</b> ".$user."<br/>";
 echo "<b>Ihr Alter:</b> ".$age."<br/>";
 echo "<b>Verheiratet?:</b> ".$married."<br/>";
 ?>
</BODY>
</HTML>
```

Datenbankanbindung

- **Was tun mit den Formulardaten, die im vorherigen Beispiel von unserem PHP-Skript eingelesen wurden?**
  - **Auf dem Bildschirm ausgeben? Ok, aber was dann?**
  - **In einer Datei speichern? Ok, aber Probleme z.B. beim gleichzeitigen Zugriff durch parallele Aufrufe …**
  - **In einer Datenbank !!!**
- **Im Folgenden erhalten Sie eine sehr eingeschränkte Einführung in relationale Datenbanksysteme**
  - **Darstellung von Datensätzen in Tabellen (hier: nur eine Tabelle)**
  - **Anfrage an eine Tabelle mit der Anfragesprache SQL**
  - **Einbindung in PHP**
- **Mehr dazu in der Vorlesung "Datenbanksysteme"**

Relationale Datenbanken

- **In sogenannten Relationalen Datenbanksystemen werden Daten in Tabellen abgelegt**
- **Ein zusammengehöriger Datensatz wird dabei in eine Zeile einer Tabelle abgelegt**
- **Spalten einer Tabelle enthalten Elemente vom gleichen Typ und haben einen Namen (= Attributname)**
- **Typen, z.B. int, varchar(n)**
- **Beispiel: Tabelle "Nutzer"**

id	name	alter	verheiratet
1	Karl	44	1
2	Sabine	28	0
3	Heinz	34	0
4	Eva	39	1

Datenbankabfragen mit SQL

- **SQL-Structured Query Language**
  - **Standard-Abfragesprache für relationale Datenbanksysteme**
  - **Bietet Sprachkonstrukte zum**
    - **Erzeugen von Datenbanken**
    - **Stellen von Anfragen an Datenbanken**
    - **Speichern eines neuen Datensatzes**
    - **Ändern oder Löschen von Datensätzen**
- **Anfragen:**
  - **SELECT <attribute> //welche Attribute (Spalten) auswählen?**
    FROM <tabelle> // von welcher Tabelle?
    WHERE <bedingung> // zusätzliche Filterbedingungen?
- **SELECT-Anfragen liefert grundsätzlich eine Tabelle zurück (d.h. eine Menge von Datensätzen)**

SQL-SELECT: Beispiele

SELECT * FROM Nutzer

Liefert alle Einträge der Tabelle mit allen Attributen

id	name	alter	verheiratet
1	Karl	44	1
2	Sabine	28	0
3	Heinz	34	0
4	Eva	39	1

SELECT name, alter
FROM Nutzer
WHERE verheiratet > 0

Liefert die Attribute name und alter der Datensätze, deren Wert für das Attribut verheiratet größer als 0 ist

name	alter
Karl	44
Eva	39

Einfügen eines neuen Datensatzes

- **Zum Einfügen von Datensätzen gibt es das SQL-Sprachkonstrukt**

INSERT INTO
<tabelle>(<attribute>)
VALUES (<werte>);

- **Beispiel:**

INSERT INTO
Nutzer (name, alter, verheiratet)
VALUES ("Olivier", 27, 0);

- **Für das Feld id muss kein Wert eingegeben werden**

id	name	alter	verheiratet
1	Karl	44	1
2	Sabine	28	0
3	Heinz	34	0
4	Eva	39	1
5	Olivier	27	0

Datenbankabfragen mit PHP

- SQL-Anfragen und -Befehle werden nicht vom PHP-Prozessor , sondern von einer Datenbank ausgeführt
- PHP muss deshalb:
  - Sich mit der Datenbank verbinden
  - Die Anfrage an die Datenbank schicken
  - Das Ergebnis (ein sogenanntes "Result Set") entgegennehmen
  - In einer Variable weiterbearbeiten
- Es gibt unterschiedliche Datenbanken (z.B. mySQL, SQLite)
- Es gibt unterschiedliche Programmerweiterungen, die die Kommunikation zwischen PHP und der Datenbank ermöglichen
- Näheres dazu erfahren Sie in der Übung / mit den Übungsmaterialien