

10. Internet-basierte Digitale Medien

10.1 Clientseitige Web-Skripte: JavaScript

10.2 Document Object Model (DOM)

10.3 Serverseitige Web-Skripte: PHP



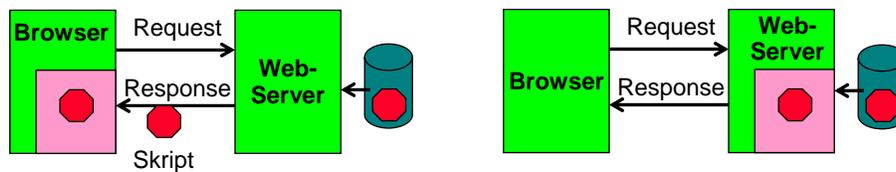
Weiterführende Literatur:

Wolfgang Dehnhardt: JavaScript, VBScript, ASP, Perl, PHP, XML:
Skriptsprachen für dynamische Webauftritte, Carl Hanser 2001

<http://www.php.net>

<http://www.selfphp.info>

Serverseitige vs. clientseitige Dynamik



- Clientseitige Dynamik:

- Browser enthält Ausführungsmaschine für Skripte
- Skript ist Teil der Antwort vom Server
- Web-Server muss Skriptsprache nicht kennen
- Beispiel: JavaScript

- Serverseitige Dynamik:

- Web-Server enthält Ausführungsmaschine für Skripte
- Skript wird vor Beantwortung der Anfrage ausgeführt und liefert HTML-Text
- Browser muss Skriptsprache nicht kennen
- Beispiel: PHP

Technologien für serverseitige Dynamik

- Common Gateway Interface (CGI):
 - Ermöglicht Aufruf beliebiger Programme beim Server (z.B. in C)
 - Programm erzeugt (schreibt) HTML-Textdatei (Response)
 - Häufig Skriptsprachen benutzt (z.B. Perl, Python)
 - Manchmal spezielle Bibliotheken für Webseiten verfügbar
- Spezielle Server-Skriptsprachen:
 - Entworfen für Einbettung in HTML
 - Plug-Ins für gängige Web-Server-Software
 - geeignet für "mittelgroße" dynamische Anwendungen
- Programmiersprachen-Einbettung in Web-Server:
 - z.B. Java *Servlets*, geschrieben in Java
 - Aufgerufen vom Server über standardisiertes API
 - Generierung von Servlets aus Skript-ähnlichen Sprachen
z.B. Java Server Pages (JSP)
 - geeignet für "große" dynamische Web-Anwendungen
(Servlets/JSP: siehe Vorlesung Medientechnik!)

Beispiel: Server-Skriptsprache PHP

- PHP:
 - **P**ersonal **H**ome **P**age Toolkit
 - **P**HP **H**ypertext **P**reprocessor
- OpenSource Entwicklung:
 - siehe www.php.net
 - lizenzfrei benutzbar
- Syntax an C angelehnt, aber mehrere Syntax-Varianten unterstützt
- Einfache Kernsprache, umfangreiche Funktionsbibliothek
 - über 500 Funktionen!
 - etwas unübersichtlich
 - spezialisiert auf Aufgaben der Webseiten-Programmierung

Voraussetzungen für praktische Experimente

- Auch bei lokalen (Ein-Rechner-)Experimenten
 - Installation eines Web-Servers
 - » OpenSource: *Apache*
 - » Microsoft *Internet Information Server*
 - Aufruf der HTML-Dateien über Web-Server (<http://...>)
- Bereitstellung und Installation der PHP-Software als Plug-In für den verwendeten Web-Server
- In den meisten praktischen Fällen: Installation eines relationalen Datenbanksystems (z.B. MySQL)
- Insider-Kürzel für bestimmte Konfigurationen (Beispiele):
 - LAMP: Linux, Apache, MySQL, PHP
 - WIMP: Windows, Internet Information Server, MySQL, PHP
 - MOXAMP: MacOS X, Apache, MySQL, PHP (hier verwendet)

Beispiel: "Hello World" in PHP und JavaScript

```
<html>
<head><TITLE>Hello World mit JavaScript</TITLE></head>
<body>
  <h1>
    <script type="text/javascript">
      document.write("Hello World!");
    </script>
  </h1>
</body>
</html>
```

JavaScript

```
<html>
<head><title>Hello World mit PHP</title></head>
<body>
  <h1>
    <?php
      echo "Hello World!";
    ?>
  </h1>
</body>
</html>
```

PHP

Einbettung von PHP in HTML

- XML-Stil (hier verwendet):
 - Analog zu *Processing Instructions* von XML

```
<?php PHP-Text ?>
```
- SGML-Stil:
 - Kurze und weit verbreitete "Urform"

```
<? PHP-Text ?>
```
- HTML-Stil:
 - Analog zur JavaScript-Einbettung

```
<script language="php"> PHP-Text </script>
```

(Lästige) Details: Syntaktische Unterschiede

- Generell stärkere Anlehnung an Shell-Skriptsprachen
 - Variablen beginnen immer mit "\$"
 - Viele UNIX-Kommandos direkt verfügbar, z.B.

```
echo "Beispiel";
```

(statt in JavaScript: `document.write("Beispiel");`)
- Verschiedene Varianten für Steueranweisungen, z.B.:

```
if (bedingung1) anw1 elseif (bedingung2) anw2 else anw3;  
if (bedingung1): anwfolge1 elseif (bedingung2): anwfolge2  
else: anwfolge3 endif;
```
- Schwach typisiert, aber geringfügig anderes Typsystem zu JavaScript
- Arrays einschließlich assoziativer Arrays, aber etwas andere Syntax und Bibliothek als in JavaScript
- PHP ist weitgehend objektorientiert, kennt Klassen und Vererbung in Java-Syntax.

Beispiel: Fibonacci-Funktion mit PHP (Version 1)

```
<body> ...
  <h2>
    <?php
      function fib($n){
        if ($n==0)
          return 0;
        else
          if ($n==1)
            return 1;
          else
            return fib($n-1)+fib($n-2);
      };
      echo "fib(3) = ", fib(3), "<br>";
      echo "fib(8) = ", fib(8), "<br>";
    ?>
  </h2>
</body>
</html>
```

Server-Skripte und Formulare

- Benutzereingaben aus Formularen
 - Müssen zuerst zum Server übertragen werden
 - Werden dann im Server-Skript ausgewertet
 - Werden dann lokal angezeigt, indem eine neue HTML-Seite generiert wird
- HTML: Attribut **action** beim Formular-Tag **<form>**
 - Spezifiziert das zur Verarbeitung der Eingabe benutzte Server-Dokument
 - Typische Beispiele:
 - » PHP-Skript
 - » Email-Versand (action=mailto:xyz@abc.com)
 - » HTML-Seite mit eingebetteten Skripten
 - Einfacher Spezialfall:
 - » Aufruf der aktuellen Seite (Neuladen)

Fibonacci-Funktion mit PHP (Version 2): Eingabeseite mit Aufruf von PHP-Skript

```
<body>
  <h1>
    Fibonacci-Funktion (Eingabe)
  </h1>
  <h2>
    Bitte Zahlwert f&uuml;r Berechnung eingeben:
    <form name="formular" action="fibonacci2b.php">
      <input type="text" name="eingabe"
        value="0"><br>
      <input type="submit" value="Berechnen">
    </form>
  </h2>
</body>
</html>
```

Datei fibonacci2a.html

Fibonacci-Funktion mit PHP (Version 2): Ergebnisseite

```
<body>
  <h1>
    Fibonacci-Funktion (Ergebnis)
  </h1>
  <h2>
    <?php
      $eingabe = $_REQUEST['eingabe'];
      function fib($n){ wie in Version 1 };
      echo "fib($eingabe) = ";
      echo fib($eingabe);
      echo "<br>";
    ?>
    <br>
    <a href="fibonacci2a.html">Neue Berechnung</a>
  </h2>
</body>
```

Datei fibonacci2b.php

Variablen, Parameterübergabe und Sicherheit

- Globales Array `$_REQUEST` zum Zugriff auf externe, beim Aufruf übergebene, Werte verwendbar
- Assoziatives Array: Auslesen einzelner Werte durch `$_REQUEST['var'];`
- Ältere PHP-Versionen (bis 4.2.0):
 - Große Sicherheitslücke durch fehlende Unterscheidung zwischen globalen (über GET/POST oder Cookies übermittelten) und lokal benutzten Variable
 - Problem tritt nur wegen des Verzichtes auf Variablendeklaration in PHP auf!
 - Angreifer können interne Variablen durch Manipulation der URL setzen (z.B. "authorization_successful"...!)
 - Altes Verhalten durch Server-Konfiguration auch in neuen Installationen einstellbar (leider weit verbreitet)

Kombination von Eingabe- und Ergebnisseite

```
<body>
  <h1>
    Fibonacci-Funktion
  </h1>
  <h2>
    <?php
      function fib($n){ wie oben };
      $eingabe = $_REQUEST['eingabe'];
      echo "fib($eingabe) = ";
      echo fib($eingabe);
      echo "<br>";
    ?>
    <br>
    Bitte Zahlwert f&uuml;r neue Berechnung eingeben:
    <form name="formular" action="fibonacci2.php">
      <input type="text" name="eingabe" value="0"><br>
      <input type="submit" value="Berechnen">
    </form>
  </h2>
</body>
```

Beispiel für Browseranzeige



Fibonacci-Funktion

fib(12) = 144

Bitte Zahlwert für neue Berechnung eingeben:

Ludwig-Maximilians-Universität München Prof. Hußmann Digitale Medien – 10 – 45

GET- und POST-Methode in HTTP

- Das Hypertext Transfer Protocol (HTTP) unterstützt zwei Methoden, Parameterwerte an aufgerufene Dokumente zu übergeben
- GET-Methode:
 - Variablenwerte werden als Bestandteil der URL codiert und übergeben:
`http://host.dom/pfad/fibonacci2.php?eingabe=12`
 - Damit können Parameterangaben auch durch Eintippen der URL gemacht werden (ohne Formular)
 - Geeignet für einfache Abfragen
- POST-Methode:
 - Variablenwerte werden nicht in der URL codiert
 - Webserver wartet auf anschließende Übertragung der Variablenwerte (Einlesen vom Standard-Eingabekanal)
 - (Etwas) schwerer von außen zu "manipulieren"
- HTML: Attribut `method` beim Formular-Tag `<form>`
 - `method="get"` (default!) oder `method="post"`

Dauerhafte Speicherung von Information

- Webseiten-Inhalt soll oft von gespeicherter Information abhängen
 - E-Commerce, E-Government, ...
 - Personalisierung von Seiten
 - Diskussionsforen
 - ...
- Serverseitige Speicherung:
 - Grosse Datenmengen (Datenbank)
 - » aber auch einfache Dateien möglich
 - Aktualisierung durch externe Programme
 - Anbindung an komplexe Programmsysteme
- Clientseitige Speicherung:
 - Geringe Datenmengen
 - Starke Einschränkungen aus Sicherheitsgründen
 - Für Identifikation etc.: "Cookies"

Cookies

- Kleine im Browser (bzw. einer vom Browser kontrollierten Datei) gespeicherte Dateneinheiten
- Cookie enthält:
 - Name (String), auch Schlüssel genannt
 - Wert (String)
 - Verfallsdatum
 - optional: Domäne, Pfadname, Sicherheitsinformation
- Übertragung zwischen Client und Server im HTTP-Protokoll
 - Zu jeder Anfrage nach einem Dokument werden alle *zugehörigen* Cookies an den Server gesandt.
- Auf ein Cookie kann nur das Programm/der Server zugreifen, der das Cookie erzeugt hat
- Clientseitige Erzeugung/Zugriff: z.B. mit JavaScript
- Serverseitige Erzeugung/Zugriff: z.B. mit PHP

Cookies in PHP (1)

```
<?php
    $key = $_REQUEST['key'];
    $val = $_REQUEST['val'];
    $tim = $_REQUEST['tim'];
    $zeit = time() + $tim * 60;
    setcookie($key, $val, $zeit);
    echo "<tt>Cookie: ", $key, "=", $val, "<br>";
    echo "g\u00fctig bis: ",
        date("d. F Y G:i:s", $zeit);
    echo "</tt><br>";
?>
```

- Benutzergegebene Parameter (rot dargestellt, gesetzt in separater HTML-Datei mit <form>):
 - Verfallszeit (hier in Minuten ab "jetzt")
 - Name
 - Wert

Cookies in PHP (2)

```
<?php
    echo "<b>Gesetzte Cookies:</b><br>\n";
    while (list($key, $wert) = each($_COOKIE))
        echo $key, "=", $wert, "<br>\n";
?>
```

- Aktuelle Cookies verfügbar über eingebaute Variable
 - codiert in assoziativem Array `$_COOKIE`
(Frühere PHP-Versionen: `$HTTP_COOKIE_VARS`)
- Verwendete Array-Funktionen:
 - `each()`: Durchläuft alle Array-Elemente
 - `list()`: Weist Array-Werte an Variablen-Tupel zu

Ein einfaches Diskussionsforum (1)

- Interaktive Eingabe von Beiträgen
- Aktuelle Anzeige aller Beiträge
- Speicherung des aktuellen Diskussionsstandes in Datei auf Server
- Nur ca. 50 Zeilen HTML+PHP !

Diskussionsforum

Neuer Beitrag:

Name:

Beitrag (1 Zeile):

Bisherige Beiträge:

3 Beiträge

Beitrag Nr. 1:

Name: Max
Beitrag: Das ist interessant.

Ein einfaches Diskussionsforum (2)

- Beispielinhalt der Datei "forum.txt":
 - Je zwei Zeilen gehören zusammen.
 - Erste Zeile: Name
 - Zweite Zeile: Inhalt

Max

Das ist interessant.

Moritz

Das ist eher langweilig.

Ein einfaches Diskussionsforum (3)

- Ausgabe der aktuell bekannten Beiträge aus Datei
- Verwendete Dateifunktion:
 - `file()`: Wandelt Dateiinhalte in String-Array
- Verwendete Arrayfunktion:
 - `count()`: Länge des Arrays

```
<h2>Bisherige Beitr&auml;ge:</h2>
<?php
    $inhalt = file("forum.txt");
    echo "<h3>", count($inhalt)/2, " Beitr&auml;ge</h3>";
    $i = 0;
    while ($i < count($inhalt)) {
        echo "<h3>Beitrag Nr. ", ($i+2)/2, " :</h3>";
        echo "<b>Name:&nbsp;</b>", $inhalt[$i++], "<br>";
        echo "<b>Beitrag:&nbsp;</b>",$inhalt[$i++], "<br>";
    }
?>
```

Ein einfaches Diskussionsforum (4)

- Code zum Erweitern der Datei entweder in separatem Skript oder in gleicher Datei wie Anzeige-Skript (hier gezeigt)
 - Abfrage, ob Eintragen nötig (Variable `$eintragen` zeigt, ob Einfügen-Schaltfläche gedrückt wurde)
- Verwendete Dateifunktionen:
 - `fopen()`, `fclose()`: Datei öffnen ("a"=append), schliessen
 - `fputs()`: Zeichenreihe schreiben

```
<?php
    $eintragen = $_REQUEST['eintragen']; $name ...; $beitrag ...;
    if ($eintragen != "" &&
        $name != "" && $beitrag != "") {
        $datei = fopen("forum.txt", "a");
        fputs($datei,$name . "\n");
        fputs($datei,$beitrag . "\n");
        fclose($datei);
    }
?>
```

Weitere Eigenschaften von Server-Skripten

- Datenbankanschluss
 - z.B. über Open Database Connectivity (ODBC)
 - Funktionen zum Öffnen, Abfragen, Manipulieren von Datenbanken und Tabellen in Datenbanken
- Reguläre Ausdrücke
 - zur flexiblen Zeichenreihen-Verarbeitung
- Erzeugung von Print-Dokumenten
 - spezielle Bibliothek zur PDF-Erzeugung in PHP
- Weitere Protokolle
 - z.B. FTP, Email-Protokolle (SMTP, POP, IMAP)
- Diverses:
 - XML-Verarbeitung
 - Bildbearbeitung
 - Rechtschreibprüfung
 - Passwortsicherheit
 - ...

Server-Skripte vs. Client-Skripte

Client-Skripte

Schnelle Reaktion
Funktion auch ohne Netzanbindung
Unabhängigkeit von Server-Software

Server-Skripte

Berechnung von Seiteninhalt aus
Benutzereingaben und anderen
äußeren Umständen

Datenhaltung auf Server
Zugriff auf zentrale Ressourcen (z.B. zur Weiterverarbeitung)
Unabhängigkeit von Browser-Software