

# Vorlesung Werkzeuge der Informatik

## **Grundlagen und Werkzeuge des WWW (Teil 2)**

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

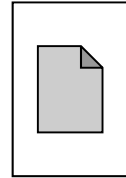
# Das Dokumentenmodell des WWW

- **Dokumente kann Referenzen (Hyperlinks) auf ein anderes Dokument enthalten**
- **Hyperlinks werden durch Browser explizit angezeigt → Auswahl durch Anklicken**
- **Auswahl eines Hyperlinks erzeugt Anforderung an den Server, auf dem das entsprechende Dokument abgelegt ist (über URL)**
- **Von dort wird es auf die Maschine des Nutzers übertragen und im Browser angezeigt**
- **Sprache für Webdokumente:  
HTML: Hypertext Markup Language**

# Hypertext Markup Language

- **Markup-Sprache zur Annotation und Strukturierung von Dokumenten**
- **HTML-Dokumente bestehen aus Kopfteil <HEAD> und Rumpf <BODY>**
- **HTML-Sprachelemente (=Tags) zum**
  - **Markieren von Überschriften, Listen, Tabellen, Formularen**
  - **Einfügen von Bildern und Animationen**
  - **Gestaltung der Darstellung im Browser (z.B. Schriften, Schriftgrößen, kursive oder fettgedruckte Anzeige, Ausrichtung, Farben, Textabsätze)**
- **Notation von HTML-Tags in spitzen Klammern, z.B.**
  - **<H1>Hello World</H1>**
- **HTML-Tags schließen in der Regel Text ein**
- **Web Browser verwendet HTML-Parser: Software, die HTML-Tags erkennt und in strukturierten Text umsetzt.**
- **Erweiterungen erlauben die Darstellung von Teilen eines Dokumentes in Form eines im Browser ausführbaren Skriptes (z.B. Javascript)**

# HTML – Ein erstes Beispieldokument



```
<HTML>
  <HEAD>
    <TITLE>Meine Homepage</TITLE>
  </HEAD>
  <BODY>
    <H1>Homepage von Jörg Müller</H1>
    Dies ist meine erste HTML-Seite.
    <p>
      Sie enthält einen Paragraphen und einen <br>Zeilenumbruch.
    </p>
    Außerdem kann man auch <b>fett</b> und <i>kursiv</i>
    schreiben.
  </BODY>
</HTML>
```

# Zeichendarstellung in HTML

- **HTML-Texte werden in Dateien in binärer Form (bytes) abgespeichert**
- **Web-Browser muss Bytes wieder in Zeichen (Buchstaben etc.) umwandeln**
  - **Zeichenkodierung, erfolgt mithilfe einer Codetabelle, die jedem erlaubten Zeichen aus dem Zeichenvorrat eine fortlaufende Nummer zuweist**
  - **Verschiedene Zeichenkodierungsschemata**
  - **z.B. UNICODE: "a" → 97, "b" → 98, "c" → 99, etc.**
- **Browser muss bei der Anzeige die gleiche Kodierung angeben, die der Editor beim Abspeichern verwendet**
  - **Spezifikation der Zeichenkodierung im Kopf des HTML-Dokumentes**

# Zeichendarstellung in HTML (2)

- **Beispiel:**

```
<head>
<meta http-equiv="content-type"
      content="text/html; charset=ISO-8859-1">
<!-- ... andere Angaben im Dateikopf ... -->
</head>
```

- **Darstellung von Sonderzeichen mittels benannter HTML-Elemente, z.B.**

- `&uuml;` für "ü", `&Uuml;` für "Ü"
- `&amp;` für "&", `&szlig;` für "ß"

- **Beispiel:**

```
J&ouml;rg M&uuml;ller trinkt im Hofbr&auml;uhaus
Bier aus Ma&szlig;kr&uuml;gen.
```

# Elemente zur Textstrukturierung

- **Bereits kennengelernt: Überschriften, Textabsätze**
- **Darstellung von Listen und Aufzählungen**

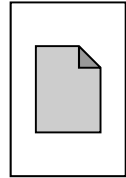
```
<ul> <!-- dies ist eine unsortierte Liste -->  
<li>Element 1</li>  
<li>Element 2</li>  
<li>Element 3</li>  
</ul>
```

```
<ol> <!-- dies ist eine nummerierte Liste -->  
<li>Element 1</li>  
<li>Element 2</li>  
<li>Element 3</li>  
</ol>
```

- **Listen können geschachtelt sein**



# Beispiel: Listen in HTML



```
<HTML>
  <HEAD><TITLE>Meine Homepage</TITLE></HEAD>
  <BODY>
    <H1>Homepage von Jörg Müller</H1>
    Berufliche Laufbahn:
      <ul>
        <li>seit 2005: Professor an der TU Clausthal</li>
        <li>1999-2006: Principal Research Scientist bei Siemens
          Corporate Technology, München</li>
        <li>Davor: <ol><li> Drei Jahre Berufserfahrung</li>
          <li> Studium und Promotion</li>
          <li> Schule und Kindergarten</li>
        </ol>
        </li>
      </ul>
    </BODY>
  </HTML>
```

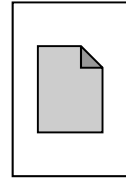
# HTML Tabellen

- Tabellen werden häufig zur Strukturierung von HTML-Dokumenten eingesetzt
- Aufbau einer Tabelle in HTML
  - Tabelle `<table>`
  - Zeile `<tr>`
  - Tabellenkopfelement `<th>`
  - Tabellendatenelement `<td>`
  - Weitere Attribute für Form (z.B. `border`)

<code>&lt;th&gt;</code>	<code>&lt;th&gt;</code>	<code>&lt;th&gt;</code>
<code>&lt;td&gt;</code>	<code>&lt;td&gt;</code>	<code>&lt;td&gt;</code>
<code>&lt;td&gt;</code>	<code>&lt;td&gt;</code>	<code>&lt;td&gt;</code>

Quelle: de.selfhtml.org

# Beispiel: Tabellen zur Strukturierung von Dokumenten

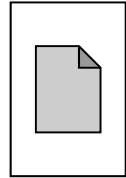


```
<HTML>
  <HEAD><TITLE>Meine Homepage</TITLE></HEAD>
  <BODY>
    <H1>Homepage von J&ouml;rger M&uuml;ller</H1>
    <table border="1">
      <tr> <td colspan="3">Berufliche Laufbahn</td></tr>
      <tr> <th>Jahr</th> <th>T&auml;tigkeit</th>
        <th>Organisation / Ort</th></tr>
      <tr> <td>seit 2005</td> <td>Professor</td>
        <td>TU Clausthal</td> </tr>
      <tr> <td>1999-2006</td> <td>Principal Research Scientist</td>
        <td>Siemens Corporate Technology, München</td> </tr>
      <tr> <td colspan="3">und so weiter</td></tr>
    </table>
  </BODY>
</HTML>
```

# HTML Formulare

- **Zweck: Eingabe (über Eingabefelder o. Auswahllisten) und Senden (über Button) von Daten vom Client zum Server**
- **Erstellen eines Formulars in HTML, dabei wird spezifiziert, was mit den eingegebenen Daten passieren soll**
- **Anwendungen:**
  - Suchmaschinen
  - Benutzeranmeldung bei einer Web-Applikation
  - Erfassen von Nutzer- und Bestelldaten in E-Shops

# Definition von Formularen: Das <form> Tag



```
<HEAD><TITLE>Meine Homepage</TITLE></HEAD>
<BODY>
  <H1>Homepage von J&ouml;rg M&uuml;ller</H1>
<p> Mit diesem Formular können Sie mit mir Kontakt aufnehmen
</p>
<form action="http://de.selfhtml.org/cgi-bin/comments.pl">
  <!-- Hier wird das Formular definiert -->
</form>
</body>
</html>
```

# Der Formularinhalt (1)

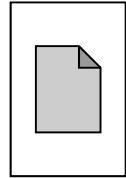
```
<form action="http://de.selfhtml.org/cgi-bin/comments.pl">
  <table border="0" cellpadding="5" cellspacing="0"
    bgcolor="#E0E0E0">
    <tr>
      <td align="right">Vorname:</td>
      <td><input name="Vorname" type="text" size="30"
        maxlength="30"> </td>
    </tr>
    <tr>
      <td align="right">Zuname:</td>
      <td><input name="Zuname" type="text" size="30"
        maxlength="40"></td>
    </tr>
    <tr>
      <td align="right" valign="top">Kommentar:</td>
      <td><textarea name="Text" rows="5" cols="50">
        </textarea></td>
    </tr>
  </table>
</form>
```

# So sieht's dann im Web Browser aus ...



## Was fehlt hier?

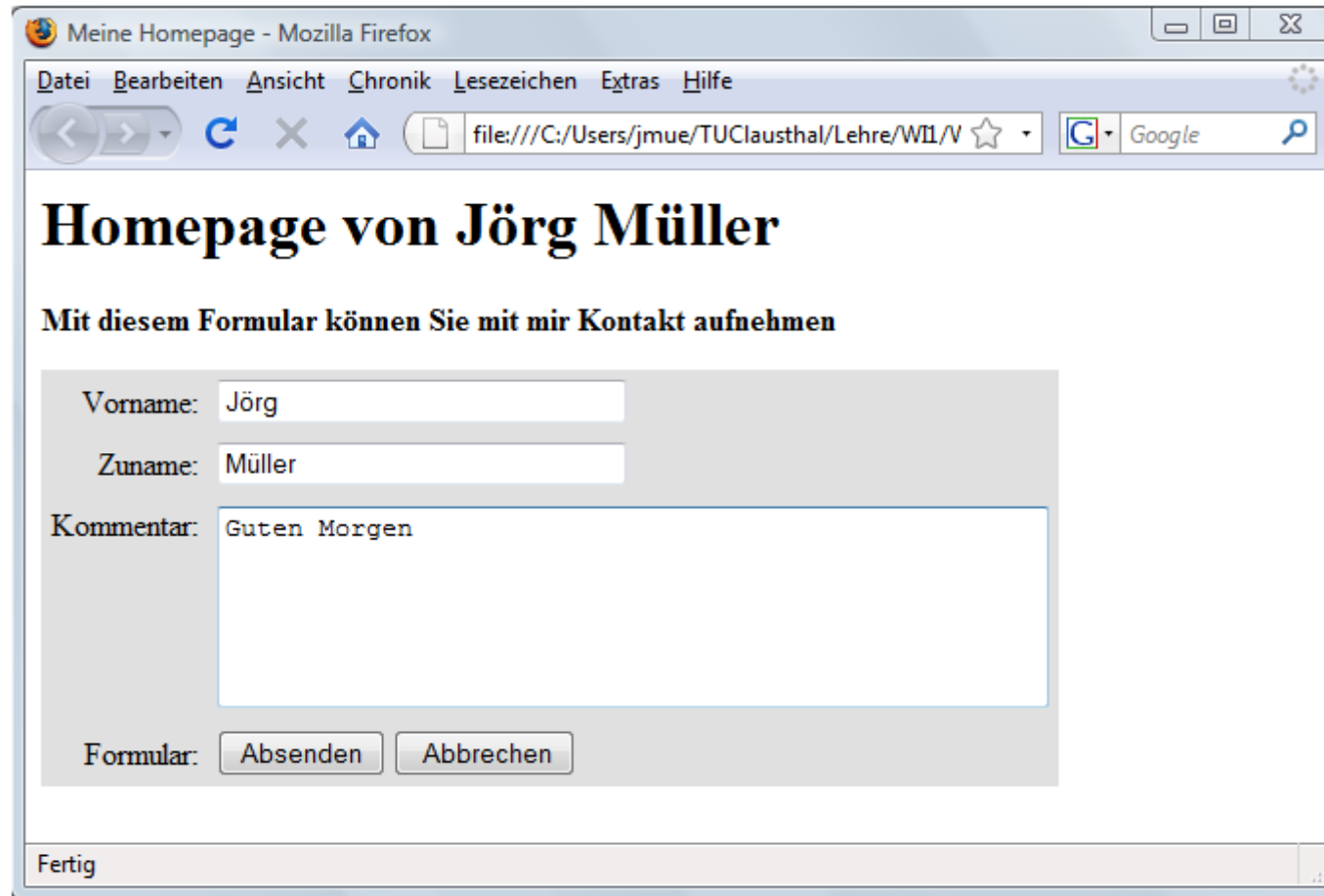
# Der Button zum Absenden!



```
<form action="http://de.selfhtml.org/cgi-bin/comments.pl">
  <table border="0" cellpadding="5" cellspacing="0"
    bgcolor="#E0E0E0">
    <tr><!-- Hier Vorname --> </tr>
    <tr><!-- Hier Zuname --> </tr>
    <tr><!-- Hier Kommentar --> </tr>
    <tr>
      <td align="right">Formular:</td>
      <td>
        <input type="submit" value=" Absenden ">
        <input type="reset" value=" Abbrechen">
      </td>
    </tr>
  </table>
</form>
```



# Das sieht dann so aus:



Und das wird übertragen:

<http://de.selfhtml.org/cgi-bin/comments.pl?Vorname=J%F6rg&Zuname=M%FCller&Text=Guten+Morgen>

# Und hier nochmal der ganze HTML Code

```
<HEAD><TITLE>Meine Homepage</TITLE></HEAD>
<BODY>
  <H1>Homepage von J&ouml;rg M&uuml;ller</H1>
  <p><b>Mit diesem Formular können Sie mit mir Kontakt aufnehmen </b></p>
  <form action="http://de.selfhtml.org/cgi-bin/comments.pl">
    <table border="0" cellpadding="5" cellspacing="0" bgcolor="#E0E0E0">
      <tr> <td align="right">Vorname:</td>
        <td> <input name="Vorname" type="text" size="30" maxlength="30"> </td> </tr>
      <tr> <td align="right">Zuname:</td>
        <td><input name="Zuname" type="text" size="30" maxlength="40"></td></tr>
      <tr> <td align="right" valign="top">Kommentar:</td>
        <td> <textarea name="Text" rows="5" cols="50"> </textarea></td> </tr>
      <tr><td align="right">Formular:</td>
        <td> <input type="submit" value=" Absenden ">
          <input type="reset" value=" Abbrechen">
        </td></tr>
    </table>
  </form>
</body>
</html>
```

# Attribute des <form> Tags

- **action:** Gibt an, was mit den Eingabedaten passieren soll, wenn das Formular abgesendet wird, z.B.
  - **Email senden:**  
`<form action="mailto:mueller@tu-clausthal.de" ...>`
  - **Programm auf dem Server aufrufen, das die Daten weiterverarbeitet**  
`<form action="http://www.tu-clausthal.de/cgi-bin/feedback.pl" ...>`
- **method:** Auswahl der http-Übertragungsmethode für die Formulardaten (s.o.)
  - **get:** Daten des ausgefüllten Formulars als Parameter an die Aufrufadresse angehängt
  - **post:** Daten des ausgefüllten Formulars werden vom Web-Server über den Standardeingabekanal (d.h. im Body des HTTP-Requests) zur Verfügung gestellt
  - **put:** Verwendet für das Upload einer Datei vom Client zum Server

# HTML-Editoren

- **In der Praxis werden zur Erstellung von HTML Editoren verwendet (z.B. Adobe GoLive, Microsoft Frontpage)**
- **Content-Management-Systeme wie Typo3 erlauben das Erstellen von HTML-Dokumenten in WYSIWYG-Form (z.B. als RTF)**
- **Für die meisten größeren Anwendungen gilt, dass HTML von einem Serverprogramm dynamisch erstellt wird, z.B.**
  - **Active Server Pages (Microsoft)**
  - **Java Server Pages**
  - **XML Dokumenten durch XML Stylesheet Transformation (XSLT)**
  - **Skriptsprachen wie Javascript oder PHP (siehe Teil 3)**

# HTML: Zusammenfassung

- **Hypertext Markup Language (HTML)**
  - einheitliches, fixes Markup-System
- **Vorteile:**
  - einfach
  - portabel
  - einheitliche Handhabung von Verweisen (Links)
- **HTML erlaubt erstmals den breiten Einsatz einheitlicher Schnittstellen zu Information und Software-Applikationen innerhalb einer Organisation**

# Nachteile von HTML

- **HTML ist nicht erweiterbar:**
  - Markup (Elemente und Attribute)
  - Verweise
- **HTML unterstützt nicht die Modellierung strukturierter Informationen (Objektmodelle, relationale Datenbanken)**
- **Integration von Software-Applikationen**
  - JavaScript, Plugins
  - Proprietäre Formate und Sprachen
- **Handhabbarkeit: unzureichende Entkopplung von Inhalt und Präsentation / Layout**
- **Ständiger Wandel der HTML Sprache**

**→ Entwicklung der eXtensible Markup Language (XML),  
siehe nächstes Modul (Prof. Hartmann)**

# Darstellung von Web-Layouts durch Cascading Stylesheets (CSS)

- **Idee: Trennung der Spezifikation von Inhalt und Form**
- **CSS = Sprache zur Definition von Formateigenschaften (z.B. Rahmen, Farben, Hintergrund, Zeichengröße, Font) einzelner HTML-Elemente**
- **Stylesheet-Format wird vom Browser angewendet, um Darstellung der Elemente zu bestimmen**
- **Für einen Inhalt (HTML) kann durch Auswahl des Stylesheets festgelegt werden, wie der Inhalt angezeigt wird**
- **Anwendungen:**
  - **Festlegung von Richtlinien für die Webdarstellung eines Unternehmens ("Corporate Look & Feel")**
  - **Optimierung der Darstellung von Inhalten für unterschiedliche Bildschirmformate und -größen (dafür besser geeignet: XSLT!)**

# CSS Sprache

- **Stylesheet besteht aus einer Menge von Formaten (Regeln)**
- **Regel hat die Form**  
`Selektor { Eigenschaft:Wert; }`
  - **Selektor wählt gewünschte Elemente aus**
  - **Eigenschaft: zu spezifizierende Eigenschaft des Elements sowie der ihr zugewiesene Wert**



# CSS Eigenschaften (Beispiele)

- **Innenabstand:**

- **erzwungener Leerraum zwischen dem Inhalt eines Elements (z.B. Paragraph <p>) und seinem eigenen Elementrand, also z.B. zwischen dem Text eines Elements und dem Rand dieses Elements, z.B.**
- **padding-bottom: Zwischenraum zwischen Elementinhalt u. unterer Elementgrenze (sichtbar bei Rahmen, Farben , ...)**
  - **Argument: Abstand (numerischer Wert, z.B. mm, cm, %, px, pt)**

- **Außenrand/Abstand allgemein**

- **margin: definiert den Rand/Abstand eines Elements für oben, rechts, unten und links (1-4 Argumente)**
  - **Ein Arg.: alle vier Ränder des Elements erhalten den gleichen Außenabstand.**
  - **Zwei Angaben: Abstand oben/unten, Abstand rechts/ links.**
  - **Drei Angaben: Abst. oben, Abst. rechts/links; Abst. unten.**
  - **Vier Angaben: oben, rechts, unten, links**

# CSS Eigenschaften: Beispiele (2)

- **Rahmen**

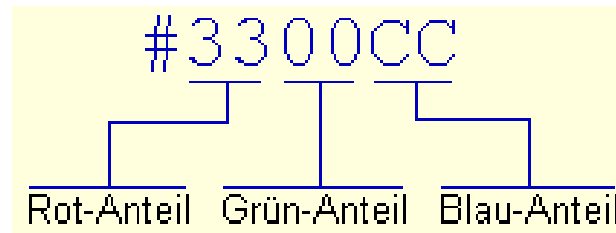
- **border**: bestimmt das Aussehen eines Rahmens um ein Element
- **Argumente**: Linienstärke, Rahmentyp und Farbe (in beliebiger Reihenfolge)
- Definierbar auch durch die Einzeleigenschaften `border-width`: `border-style`: und `border-color`:

- **Schriftformatierung**

- **Angaben zu Schriftarten, Schriftgrößen, Schriftfarben, Schriftgewicht, Zeichen- und Wortabständen**
- **Sinnvoll für alle Elemente, die Text enthalten können**
- **Eigenschaften z.B.**
  - `color`: (Textfarbe), `font-size`: Schriftgröße
  - `font-family`: Schriftart (z.B. Arial, Helvetica, Times Roman) oder Schriftfamilie (z.B. serif, sans-serif)

# Farbangaben

- **CSS Farbangaben sind möglich gemäß den Regeln für Farbangaben in HTML**
- **Angabe alternativ**
  - **durch Angabe der RGB-Werte der gewünschten Farbe in Hexadezimalform (RGB = Rot/Grün/Blau-Wert der Farbe)**



Quelle: selfhtml.org

- **durch Angabe eines Farbnamens**

```
<body bgcolor="black"> <!-- schwarzer Dateihintergrund -->  
<table bgcolor="aqua"> <!-- hellblauer Tabellenhintergrund -->  
<hr color="red"> <!-- rote Trennlinie -->
```

# CSS Regeln: Beispiele

```
h1 { border-bottom:4px double blue;  
padding-bottom:4px;  
margin:0px; }
```

```
h2 { border-top:1px solid blue;  
margin:0px; }
```

```
p { color:blue;  
font-size:200%; }
```

# Definition eines CSS in einem HTML-Dokument

## 1. Definition im Header einer Datei durch `<style>` Tag

- **Beispiel:**

```
<html> <head>
<title>Titel der Datei</title>
<style type="text/css">
<!-- ... Hier werden die Formate definiert ... -->
</style>
</head> <body> </body> </html>
```

## 2. Definition in einer eigenen Stylesheet-Datei

```
<html> <head>
<title>Titel der Datei</title>
<link rel="stylesheet" type="text/css" href="formate.css">
<style type="text/css">
<!-- ... hier sind dateispezifische Formate erlaubt ... */
</style>
</head> <body> </body> </html>
```

## 3. Definition in einem HTML-Element

```
<h1 style="[...element-spezifische Formate]">Ein Satz</h1>
```