

# Organisatorisches 2.0

- ▶ Änderung der Prüfungsleistung:
  - ▶ Vorher: **Klausur** 60 Minuten (60 Punkte)
  - ▶ Jetzt: **Referat**
- ▶ **30 Punkte für eine Klausur am Ende des Semesters**
- ▶ **30 Punkte für das Projekt**

# Webanwendungen

Vorlesung - Hochschule Mannheim

## HTML

# Inhaltsverzeichnis

- ▶ [Grundstruktur](#)
- ▶ [Formatierung](#)
- ▶ [HTML Formulare](#)
- ▶ [HTML5](#)

# HTML

- ▶ *HyperText Markup Language (HTML)*
  - ▶ eine (von vielen) Anwendung von SGML
  - ▶ *Auszeichnungssprache (markup language)*
- ▶ markiert und zeichnet Bestandteile eines Dokuments an
- ▶ Web-Browser setzen Auszeichnung in visuelle Darstellung um
  - ▶ Verweise auf andere Dokumente (*hypertext*)
- ▶ Verknüpfungen (Hyper Links) zu anderen Dokumenten
- ▶ Verweise auf Stellen im selben Dokument
  - ▶ Text-Dateien (kein Binärformat)



Tim Berners-Lee  
Quelle: Wikimedia, Enrique Dans

# Entwicklungsgeschichte

- ▶ Standardisierungsgremium W3C (<http://www.w3.org>)
  - ▶ Standardisierungsprozess langsamer als Browser
  - ▶ inkompatible Erweiterungen der Hersteller

Version	Datum	Features
2.0	Nov 95	RFC 1866
3.2	Mai 96	Tabellen, Datei-Upload, physische Formatierung
4.0	Jan 98	Frames, CSS, Skript
4.01	Dez 99	minimale Korrekturen zu HTML 4.0
XHTML		
1.0	Jan 00	Recommendation
1.1	Nov 10	Recommend. 2nd Ed.
HTML		
5	Mai 11	Working Draft

# Aktuelle HTML-Standards

- ▶ Aktueller Standard: **HTML 4.01** (von 1997)
- ▶ sauberes Konzept für HTML
- ▶ etliche Tags und Attribute sind "deprecated" (missbilligt)
- ▶ verschiedene Varianten für den Übergang (strict, transitional und frameset)
  - ▶ **XHTML 1.1** (von 2001) definiert HTML konform zu XML
  - ▶ **XHTML 2.0** wurde nie fertiggestellt (eingestampft)
  - ▶ Zukunft: **HTML5** (fertig 2022)
- ▶ ist „work in progress“
- ▶ viele Erweiterungen im Bereich Multimedia

# Grundstruktur

# Eine erste HTML-Seite

```
<!DOCTYPE html>
```

Deklaration des Dokumententyps – Hier HTML5

```
<html lang="de">
```

Wurzelelement und Sprachfestlegung

```
<head>
```

```
<meta http-equiv="Content-Type"  
      content="text/html; charset=UTF-8">
```

Zeichensatz und MIME-Typ

```
<title>Meine erste Seite</title>
```

Titel für das Browserfenster

```
</head>
```

```
<body>
```

```
<p>Ein Absatz</p>
```

Inhalt der Seite

```
</body>
```

```
</html>
```



# Dokumententypen

- ▶ HTML 4.01
- ▶ **strict** - strenge Variante von HTML 4.01, die viele alte Zöpfe abschneidet  
`<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">`
- ▶ **loose (transitional)** - erlaubt die Verwendung von abgeschafften (deprecated) Tags und soll den Übergang von HTML 3 zu 4 erleichtern  
`<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">`
- ▶ **frameset** - wie loose aber noch mit Unterstützung für Frames  
`<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"  
"http://www.w3.org/TR/html4/frameset.dtd">`
- ▶ HTML5
- ▶ nur noch eine einzige, einfache doctype-Deklaration  
`<!DOCTYPE html>`

# Zeichenkodierung (encoding)

- ▶ *Zeichenkodierung* legt fest, als welche Bytefolge ein Zeichen abgelegt wird
  - ▶ Unterschiedliche Zeichenkodierungen sind im Einsatz, z. B.
  - ▶ Codepage 850      Ä entspricht 8E
  - ▶ ISO-8859-1      Ä entspricht C4
  - ▶ Unicode (UCS2)    Ä entspricht 00 C4
  - ▶ UTF-8            Ä entspricht C3 84
  - ▶ Zeichenkodierung ist eine Eigenschaft der Datei und wird im Header dokumentiert

# Zeichensatz (Font)

- ▶ Ein **Zeichensatz (Font)** definiert das Aussehen eines Zeichens
  - ▶ Helvetica Neue      ÄÖÜabc123
  - ▶ Times New Roman      ÄÖÜabc123
  - ▶ Palatino      ÄÖÜabc123
  - ▶ Comic Sans      ÄÖÜabc123
- ▶ In HTML können über CSS verschiedene Fonts innerhalb eines Dokuments genutzt werden

# Regeln für HTML-Dokumente

- ▶ Formatierung des Dokumentes ist irrelevant
- ▶ Leerzeichen, Tabulator und Zeilenvorschub sind nur Trenner
- ▶ Einrückung dient nur Lesbarkeit, wird vom Browser ignoriert
- ▶ Absätze müssen durch spezielle Tags gekennzeichnet werden
  - ▶ Kommentare analog zum XML  
`<!-- Ein Kommentar -->`
  - ▶ Bestimmte Sonderzeichen müssen kodiert werden
    - < als &lt;
    - > als &gt;
    - & als &amp;
    - " als &quot;

# HTML-Entitäten

- ▶ HTML-Dokumente in ASCII-Codierung (7-Bit) können auch Sonderzeichen enthalten
  - ▶ `&auml;` und `&Auml;` für ä und Ä
  - ▶ `&ouml;` und `&Ouml;` für ö und Ö
  - ▶ `&uuml;` und `&Uuml;` für ü und Ü
  - ▶ `&szlig;` für ß
  - ▶ ...

# Tags in HTML

- ▶ *Tags* markieren Abschnitte im Text
  - ▶ Name in spitzen Klammern, z.B. `<h2>`
  - ▶ Schließendes Tag wird durch / gekennzeichnet, z.B. `</h2>`
  - ▶ Gleicher Name für öffnendes und schließendes Tag  
`<h2>Überschrift</h2>`
  - ▶ Tags können verschachtelt werden  
`<h2><em>Überschrift</em> mit Text</h2>`

# Leere Tags

- ▶ Es gibt Tags ohne Inhalt (leere Elemente), z.B. `<br>`
- ▶ Damit HTML eine Baumstruktur behält, muss es zu jedem öffnenden ein schließendes Tag geben
  - ▶ Tag kann sich selber schließen, z.B. `<br/>`
  - ▶ Tag als leeres Element schreiben, z.B. `<br><br/>`

# Attribute in HTML

- ▶ Öffnende Tags können Attribute enthalten
  - ▶ Paar aus einem Namen und einen Wert
  - ▶ Wert wird in Anführungszeichen geschrieben (XHTML)  
`<h2 id="level2">Überschrift</h2>`
  - ▶ *Universalattribute* sind bei jedem Tag möglich
- ▶ `id` - eindeutige ID für das Element (für Skripte)
- ▶ `class` - Name der CSS-Klasse im Stylesheet
- ▶ `style` - eingebettete Style Sheet Attribute
- ▶ `title` - Erläuterung zum Element
- ▶ `lang, dir` - Sprache und Laufrichtung des Textes



# Formatierung

# Strukturierung von Text

- ▶ Überschriften
  - ▶ `<h1>` - Überschrift der höchsten Gliederungsebene
  - ▶ `<h6>` - Überschrift der niedrigsten Gliederungsebene
- ▶ Abschnitte
  - ▶ `<p>` - Textabsatz
  - ▶ `<div>` - allgemeiner Block (für CSS)
  - ▶ `<span>` - Inline-Element für CSS (definiert keinen Block)

# Strukturierung von Text

- ▶ Aufzählungen
  - ▶ `<ol>` - Nummerierte Aufzählung
  - ▶ `<ul>` - Aufzählung ohne Numerierung
  - ▶ `<li>` - Element einer Aufzählung
- ▶ Zeilenumbruch erzwingen und verhindern
  - ▶ `<br/>` - expliziter Zeilenumbruch
  - ▶ `&nbsp;` - geschütztes Leerzeichen (verhindert Zeilenumbruch)
  - ▶ `&shy;` - Bindestrich bei Bedarf (soft hyphen)

# Beispiele für logische Auszeichnungen

`<h1>Überschrift auf Ebene 1</h1>`

`<h2>Unterüberschrift</h2>`

`<p>Ein Absatz mit etwas Text, wobei hier<br/>ein Zeilenumbruch  
erzwungen wurde.</p>`

`<h3>Nummerierte Liste</h3>`

`<ol>`

`<li>Erstens</li><li>Zweitens</li><li>Drittens</li>`

`</ol>`

`<h3>Nicht-nummerierte Liste</h3>`

`<ul>`

`<li>Erstens</li><li>Zweitens</li><li>Drittens</li>`

`</ul>`

# Tags zur physischen Formatierung

- ▶ Definieren das physische Erscheinungsbild
- ▶ Sollten vermieden werden
- ▶ Beispiele
- ▶ `<b>` - fette Schrift (bold)
- ▶ `<i>` - kursive Schrift (italic)
- ▶ `<tt>` - nichtproportionale Schrift (teletype)
- ▶ `<big>` - Schrift größer als normal
- ▶ `<small>` - Schrift kleiner als normal
- ▶ `<sup>` - Schrift hochgestellt
- ▶ `<sub>` - Schrift tiefgestellt
- ▶ `<pre>` - vorformatierter Text

- **fette Schrift (bold)**
- *kursive Schrift (italic)*
- nichtproportionale Schrift (teletype)
- Schrift größer als normal
- Schrift kleiner als normal
- Schrift<sup>hochgestellt</sup>
- Schrift<sub>tiefgestellt</sub>
- vorformatierter Text

# Tags für die visuelle Gestaltung

- ▶ Tags zur visuellen Gestaltung sollten nicht mehr benutzt werden, sondern stattdessen CSS
  - ▶ Farbangaben: `background`, `bgcolor`, `text`, `link`, `alink`, `vlink`
  - ▶ Schrift: `<font>`, `<basefont>`, `compact`, `strike`, `s`, `u`
  - ▶ Ausrichtung: `align`, `nowrap`, `<center>`
  - ▶ Größe: `size`, `width`, `height`
  - ▶ Rand: `hspace`, `vspace`, `border`
- ▶ HTML5 strict verbietet diese Tags und Attribute

# Tabellen

- ▶ Definition von Tabellen in HTML
  - ▶ `<table>` - definiert Tabelle
  - ▶ `<tr>` - eine Zeile der Tabelle (*table row*)
  - ▶ `<th>` - eine Zelle mit Tabellenüberschrift (*table header*)
  - ▶ `<td>` - eine Zelle mit Tabellen-Daten (*table data*)
  - ▶ `colspan=...` - Spalten verbinden
  - ▶ `rowspan=...` - Zeilen verbinden
  - ▶ `width=...` - Breite einer Spalte oder der Tabelle angeben
- ▶ Sollten nicht für Layoutzwecke missbraucht werden

# Beispiel für Tabellen

```
<table border="1">  
  <tr>  
    <th>Kopfzelle: 1Z1S</th>  
    <th>Kopfzelle: 1Z2S</th>  
  </tr>  
  <tr>  
    <td>Datenzelle: 2Z1S</td>  
    <td>Datenzelle: 2Z2S</td>  
  </tr>  
  <tr>  
    <td>Datenzelle: 3Z1S</td>  
    <td>Datenzelle: 3Z2S</td>  
  </tr>  
</table>
```

Kopfzelle: 1Z1S	Kopfzelle: 1Z2S
Datenzelle: 2Z1S	Datenzelle: 2Z2S
Datenzelle: 3Z1S	Datenzelle: 3Z1S



# Beispiel für Tabellen

```
<table border="1" width="20%">
  <tr>
    <th width="60%">K1</th>
    <th width="20%">K2</th>
    <th width="20%">K3</th>
  </tr>
  <tr>
    <td colspan="2">A</td>
    <td>B</td>
  </tr>
  <tr>
    <td>C</td>
    <td>D</td>
    <td rowspan="2">E</td>
  </tr>
  <tr>
    <td>F</td>
    <td>G</td>
  </tr>
</table>
```

K1	K2	K3
A		B
C	D	E
F	G	

# Do it Yourself

- Erstellen Sies eine Tabelle in HTML

Gruppe: The Masters			
Vorname	Name	Studiengang	Matrikelnummer
Igor	Master	Informatik	11234
Marcell	Kirby	Medieninformatik	10324
Thomas	Smits	Informatik	11111

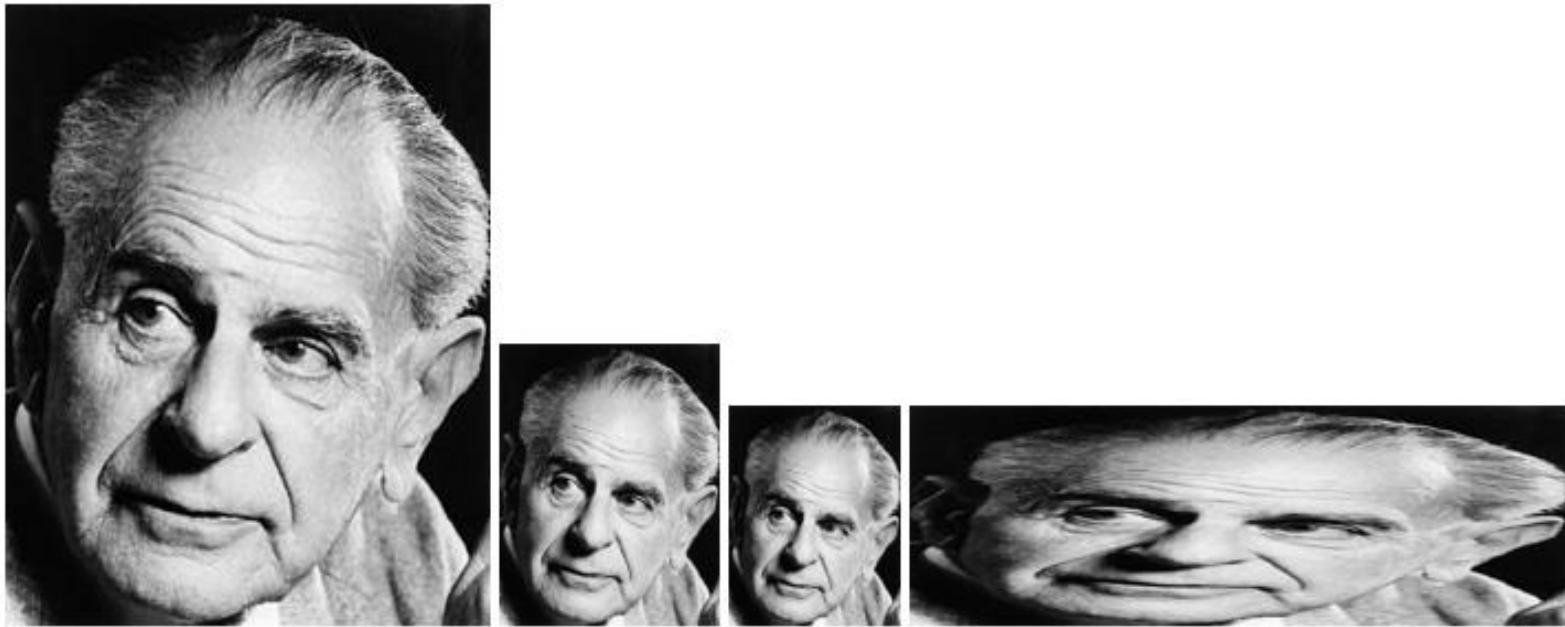
# Grafiken

- ▶ Einbinden von Grafiken mit `<img>`-Tag
  - ▶ `src` - URL von der das Bild geladen werden kann
  - ▶ `alt` - Alternativer Text für Sehbehinderte und Textbrowser
  - ▶ `width, height` - Abmessungen der Grafik
- ▶ Weitere Attribute (besser per CSS festlegen)
  - ▶ `border` - Rahmen anzeigen oder nicht
  - ▶ `hspace, vspace` - Abstand zum umgebenden Text
  - ▶ `align` - Ausrichtung und Textumfluss

# Beispiel für Grafikeinbindung

```
  
  
  

```



# Meta-Tags

- ▶ *Meta-Tags* liefern Daten, die nicht angezeigt werden
  - ▶ Anweisungen für Server
  - ▶ Anweisungen für Browser
  - ▶ Anweisungen für Suchmaschinen
- ▶ Beispiele
  - ▶ `<meta name="description" content="Web-Anwendungen"/>`
  - ▶ `<meta name="author" content="M.Kraus"/>`
  - ▶ `<meta name="keywords" content="HTML, JavaScript, PHP"/>`
  - ▶ `<meta name="robots" content="noindex" />`
  - ▶ `<meta name="date" content="2014-03-26" />`
  - ▶ `<meta name="language" content="de" />`

# Anwendungen für Hyperlinks

- ▶ Einsatzmöglichkeiten für Hyperlinks
  - ▶ Querverweis (siehe ...)
  - ▶ Blättern (nächste Seite / vorige Seite)
  - ▶ Inhaltsverzeichnis
  - ▶ Stichwortverzeichnis
  - ▶ freie Navigation durch die Dokumentstruktur
  - ▶ Download einer Datei
  - ▶ ...

# Aufbau eines Hyperlinks

- ▶ Hyperlinks werden über das `<a>`-Tag realisiert
- ▶ Format: `<a href="URL">Text</a>`
- ▶ URL: `protokoll://server:port/verzeichnis/datei#anker`
- ▶ Absolute Verweise
  - `<a href="http://www.hs-mannheim.de/php/docu.html">Dokumentation</a>`
- ▶ enthalten vollständige URL (inklusive Server)
- ▶ können auch auf externe Server zeigen
- ▶ Relative Verweise
- ▶ sind relativ zur aktuellen URL
  - `<a href="php/documentation.html">Dokumentation</a>`
- ▶ sind relativ zum aktuellen Server
  - `<a href="/php/documentation.html">Dokumentation</a>`

# Beispiele für Hyperlinks

- ▶ `<a href="index.html">Startseite</a>`
- ▶ `<a href="php/documentation.html">In einem Unterverzeichnis</a>`
- ▶ `<a href=" ../index.html">Übergeordnetes Verzeichnis</a>`
- ▶ `<a href="http://www.php.net/downloads.php">Download PHP</a>`
- ▶ `<a href="mailto:m.kraus@hs-mannheim.de">Mail an Martina Kraus</a>`
- ▶ `<a href="ftp://ftp.uni-stuttgart.de/setup.zip">Download per FTP</a>`
- ▶ `<a href="file:///Users/martina/WAW/tabelle.html">lokale Datei</a>`



# Beispiel für Verweise in derselben Datei

```
<a href="#bach">Bach</a><br/>
```

```
<a href="#interpunkt">Interpunkt</a><br/>
```

```
<a href="#interpunkt">Oxmox</a><br/>
```

```
<a href="#huegel">Hügel</a>
```

```
<p id="bergen">Weit hinten, hinter den Wortbergen, fern der Länder Vokalien  
und Konsonantien leben die Blindtexte.</p>
```

```
<p id="bach">Ein kleines Bächlein namens Duden fließt durch ihren Ort und versorgt sie  
mit den nötigen Regelialien.</p>
```

```
<p id="interpunkt">Nicht einmal von der allmächtigen Interpunktion werden die Blindtexte  
beherrscht – ein geradezu unorthographisches Leben. Eines Tages aber beschloß eine  
kleine Zeile Blindtext, ihr Name war Lorem Ipsum, hinaus zu gehen in die weite  
Grammatik.</p>
```

```
<p id="huegel">Als es die ersten Hügel des Kursivgebirges erklommen hatte, warf es einen  
letzten Blick zurück auf die Skyline seiner Heimatstadt Buchstabhausen, die Headline von  
Alphabetdorf und die Subline seiner eigenen Straße, der Zeilengasse. Unterwegs traf es  
eine Copy. Die Copy warnte das  
Blindtextchen, da, wo sie herkäme wäre sie</p>
```

# HTML-Formulare

# HTML-Formulare

- ▶ *HTML-Formulare* dienen der Eingabe von Daten
  - ▶ Daten werden an den Server übermittelt
  - ▶ Zwei HTTP-Methoden, um Daten zu übertragen
- ▶ GET - Daten werden als Teil der Request-URL übertragen
- ▶ POST - Daten werden im Request-Body übertragen
  - ▶ Bereiche in der HTML-Seite werden mit `<form>`-Tag markiert
- ▶ action - URL, die die Daten erhalten soll
- ▶ method - HTTP-Methode für die Datenübertragung (GET | POST)
- ▶ accept-charset - Zeichensatz für die übertragenen Daten
  - ▶ Innerhalb des Formulars kommen verschiedene Steuerelemente zum Einsatz

# Einzeilige Textfelder

- ▶ `<input type="text" .../>`
  - ▶ `name` - Name des Feldes (wird vom Server ausgewertet)
  - ▶ `value` - Eingegebener Wert und Vorbelegung
  - ▶ `readonly` - Nur Anzeige, keine Eingabe
  - ▶ `size` - Angezeigte Größe (in Zeichen)
  - ▶ `maxlength` - Maximale Anzahl der Zeichen in Eingabe
- ▶ `<input type="password" .../>`
  - ▶ wie Text nur werden die Zeichen nicht angezeigt

# Beispiel: einzeilige Textfelder

Einzeiliges Textfeld

```
<input type="text" size="30" maxlength="40" value="Hugo"/>
```

Readonly-Feld

```
<input type="text" size="30" maxlength="40"  
value="unveränderlich" readonly="readonly"/>
```

Passwort-Feld

```
<input type="password" size="12" maxlength="40"/>
```

Einzeiliges Textfeld

Readonly-Feld

Passwort-Feld

# Mehrzeilige Textfelder

- ▶ `<textarea ... >Text</textarea>`
  - ▶ Mehrzeiliges Textfeld (bei Bedarf mit Scrollbalken)
  - ▶ `name` - Name des Feldes (wird vom Server ausgewertet)
  - ▶ `cols` - Anzahl der Spalten
  - ▶ `rows` - Anzahl der Zeilen
- ▶ Mehrzeiliges Textfeld`<br/>`
- ▶ `<textarea name="feedback" cols="40" rows="5">Ein kleines Bächlein namens Duden fließt durch ihren Ort und versorgt sie mit den nötigen Regelialien.</textarea>`

## Mehrzeiliges Textfeld

Ein kleines Bächlein namens Duden fließt durch ihren Ort und versorgt sie mit den nötigen Regelialien.

# Auswahllisten

- ▶ Auswahllisten und Drop-Down-Boxen mit `<select>`-Tag
  - ▶ `<select>`-Tag umschließt die Optionen
  - ▶ `name` - Name des Feldes
  - ▶ `size` - Anzahl der angezeigten Zeilen (nur bei Auswahlliste)
  - ▶ `multiple` - Mehrfachauswahl zulassen (wandelt Drop-Down in Auswahlliste um)
    - ▶ `<option>`-Elemente zeigen mögliche Optionen an
  - ▶ `selected` - Eintrag ist ausgewählt
  - ▶ `value` - Wert, der bei Auswahl des Eintrages übertragen werden soll

# Beispiel: Auswahlliste / Drop-Down

Auswahlliste<br/>

```
<select name="fruit" size="2" multiple="multiple">
  <option value="apple" selected="selected">Äpfel</option>
  <option value="pear">Birnen</option>
  <option value="orange">Orangen</option>
  <option value="kiwi" selected="selected">Kiwi</option>
</select>
```

Drop-Down<br/>

```
<select name="fruit" size="1">
  <option value="apple">Äpfel</option>
  <option value="pear">Birnen</option>
  <option value="orange" selected="selected">
    Orangen</option>
  <option value="kiwi">Kiwi</option>
</select>
```

Auswahlliste



Drop-Down





# Radiobuttons und Checkboxes

- ▶ *Radiobuttons* - Gruppe von Knöpfen bei denen nur einer ausgewählt werden kann
- ▶ *name* - Gruppierung zusammengehöriger Knöpfe
- ▶ *value* - Wert, der übertragen wird
- ▶ *checked* - Button ist aktiviert
- ▶ *Checkboxen* - Gruppe von Eingaben, bei denen beliebig viele ausgewählt werden dürfen
- ▶ *name* - Name der Gruppe
- ▶ *value* - Wert, der bei markierter Box übertragen wird
- ▶ *checked* - Box ist aktiviert

# Beispiel: Radio-Button

Radio-Button<br/>

```
<input type="radio" name="fruit"  
      value="apple"/>Äpfel<br/>
```

```
<input type="radio" name="fruit"  
      value="pear"/>Birnen<br/>
```

```
<input type="radio" name="fruit" checked="checked"  
      value="orange"/>Orangen<br/>
```

```
<input type="radio" name="fruit"  
      value="kiwi"/>Kiwi<br/>
```

Radio-Button

☐ Äpfel

☐ Birnen

☒ Orangen

☐ Kiwi

# Beispiel: Checkboxes

Checkboxes<br/>

```
<input type="checkbox" name="fruit" value="apple" checked="checked"/>Äpfel<br/>
```

```
<input type="checkbox" name="fruit" value="pear"/>Birnen<br/>
```

```
<input type="checkbox" name="fruit" value="orange" checked="checked"/>Orangen<br/>
```

```
<input type="checkbox" name="fruit" value="kiwi"/>Kiwi<br/>
```

Checkboxes

☒ Apfel

☐ Birnen

☒ Orangen

☐ Kiwi

# Schaltflächen (Button) in Formularen

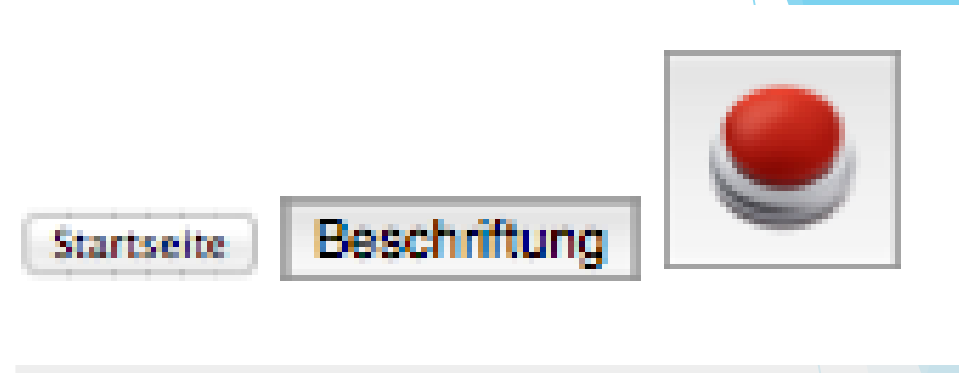
- ▶ Absenden von Formularen
  - ▶ Standard-Knopf mit `<input type="submit" .../>`
  - ▶ Selbst gestalteter Knopf mit `<button type="submit" ...>`
    - ▶ Auslösen von JavaScript-Ereignissen
  - ▶ Standard-Knopf mit `<input type="button" .../>`
  - ▶ Selbst gestalteter Knopf mit `<button type="button" ...>`
    - ▶ Attribute
      - ▶ **name** - Name des Knopfes
      - ▶ **value** - Übertragener Wert und Beschriftung (bei Standard-Knöpfen)

# Beispiel: Schaltflächen

```
<input type="button" name="Start" value="Startseite"/>
```

```
<button type="button" name="Start" value="Startseite">  
  Beschriftung  
</button>
```

```
<button type="button" name="Start" value="Startseite">  
    
</button>
```



```
<input type="submit" name="start" value="Startseite"/>
```

```
<button type="submit" name="Start" value="Startseite">  
  Beschriftung  
</button>
```

```
<button type="submit" name="Start" value="Startseite">  
    
</button>
```

# Versteckte Formularfelder

- ▶ `<input type="hidden" .../>`
  - ▶ Formularfelder werden nicht angezeigt
  - ▶ Daten werden zusammen mit anderen Feldern an den Server gesendet
  - ▶ Attribute
    - ▶ `name` - Name des Feldes
    - ▶ `value` - Wert
  - ▶ Achtung: Es gibt Tools, um die Felder im Browser zu manipulieren

# Beschriftung von Formularelementen

- ▶ `<label for="id" ...>`
  - ▶ Verknüpft Beschriftung mit Formularfelde
  - ▶ Wert im Attribut `for` bezieht sich auf die ID eines Feldes
  - ▶ Anwendbar auf `<input>`, `<select>` und `<textarea>`
  - ▶ Beim Klicken auf den (zugeordneten) Text wird das Eingabefeld selektiert bzw. die Checkbox aktiviert
  - ▶ Wichtige Unterstützung für Screenreadern

# HTML5



# Was ist HTML5?

- ▶ Implementierung der **Browserhersteller bestimmt** die HTML-Praxis (Feature ohne Browserunterstützung sind wertlos)
- ▶ W3C liefert **1997** mit **HTML 4.01** den letzten Standard
- ▶ W3C versucht mit **XHTML 2.0** den ganz großen Wurf
- ▶ Browserhersteller wollen XHTML nicht und machen einen Gegenentwurf mit HTML5
- ▶ **Browserhersteller ignorieren XHTML**, implementieren HTML5-Features und bestimmen damit die HTML-Praxis
- ▶ W3C gibt XHTML auf und eröffnet eigene HTML5-Arbeitsgruppe



# Wer standardisiert HTML5?

- ▶ Um HTML5 kümmern sich zwei Gremien
  - ▶ *Web Hypertext Application Technology Working Group (WHATWG)*
  - ▶ Initiator von HTML5
  - ▶ Zusammenschluss von Browserherstellern (z. B. Apple, Mozilla, Opera)
  - ▶ Orientiert sich stark an den Implementierungen der Browser
    - ▶ *World Wide Web Consortium (W3C)*
  - ▶ Ist später auf den HTML5-Zug aufgesprungen
  - ▶ Zusammenschluss vieler Organisationen und Einzelpersonen
  - ▶ Geleitet von Tim Berners-Lee
- ▶ HTML5 ist ein lebender Standard - Fertigstellung geplant für 2022 (sic!)

# Grundlagen

# HTML5-Präambel

- ▶ Ein HTML5-Dokument hat nur noch eine minimale Dokumententyp-Deklaration  
`<!DOCTYPE html>`
- ▶ Die Zeichenkodierung kann direkt angegeben werden  
`<meta charset="UTF-8">`  
statt  
`<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">`

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
```

# HTML-Syntax

- ▶ HTML-Syntax: `<!DOCTYPE html>`
- ▶ Tags und Attribute können groß oder klein geschrieben werden
- ▶ schließende Tags dürfen fehlen
- ▶ Anführungszeichen dürfen bei Attributen fehlen
- ▶ Attribute ohne Wert sind zulässig

```
<!DOCTYPE html>
<meta charset="UTF-8">
<title>Hallo</title>
<body>
  <p>Hello World.
  <FORM METHOD=POST>
    <input type=CHECKBOX checked>Item 1
    <INPUT type=checkbox>Item 2
  </FORM>
```

# Kategorisierung der Elemente

- ▶ HTML5 kategorisiert die HTML-Elemente
  - ▶ *Metadata* - Metadaten des Dokuments (z.B. `<base>`, `<title>`)
  - ▶ *Flow* - Inhalt des Dokuments (z.B. `<p>`, `<img>`)
  - ▶ *Sectioning* - Einteilung des Dokuments in Abschnitte (neu) (z.B. `<section>`, `<nav>`)
  - ▶ *Heading* - Überschriften (z.B. `<h1>`, `<h2>`, `<hgroup>`)
  - ▶ *Phrasing* - Auszeichnung auf der Textebene (z.B. `<span>`)
  - ▶ *Embedded* - Einbinden von Inhalten (z.B. `<iframe>`, `<video>`)
  - ▶ *Interactive* - Elemente für die Nutzerinteraktion (z.B. `<a>`, `<input>`)

# Semantisches HTML5

# Mängel von HTML 4.01

- ▶ HTML 4.01 kennt zur Strukturierung des Dokuments nur sechs Überschriftebenen `<h1>` - `<h6>`
- ▶ Webdesigner machten exzessiv vom `<div>`-Tag gebrauch, um das Dokument zu strukturieren und auszuzeichnen
- ▶ trägt aber keine semantische Bedeutung
- ▶ taucht in verschiedenen Kontexten auf
- ▶ Eine ganze Reihe von Tags dienten (trotz CSS) der Beschreibung der Präsentation (z.B. `<center>`)



# Neue Strukturierungsmöglichkeiten

- ▶ HTML5 führt neue Tags zur Strukturierung ein
  - ▶ `<section>` - unterteilt das Dokument in Sinnabschnitte
  - ▶ kann beliebig ineinander verschachtelt werden
  - ▶ in jedem Abschnitt können Überschriften-Tags neu verwendet werden
  - ▶ `<header>` - markiert einführende Inhalte
  - ▶ `<footer>` - markiert den Fußbereich von Abschnitten
  - ▶ `<nav>` - Darstellung von Navigationselementen
  - ▶ `<aside>` - Ergänzungen zu dem umgebenden Element (z.B. Seitenleiste, Kästen etc.)
  - ▶ `<article>` - Abgeschlossene Inhaltsabschnitte (z.B. Blogposts)

# Beispiel: <section>

```
<h1>Hauptüberschrift</h1>
```

```
<section>
```

```
  <h1>Kapitel 1</h1>
```

```
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
```

```
  <p>Nulla consequat massa quis enim. Donec pede justo.</p>
```

```
</section>
```

```
<section>
```

```
  <h1>Kapitel 2</h1>
```

```
  <p>Nullam dictum felis eu pede mollis pretium. Integer tincidunt.</p>
```

```
  <p>Cras dapibus. Vivamus elementum semper nisi.</p>
```

```
</section>
```

## Hauptüberschrift

### Kapitel 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Nulla consequat massa quis enim. Donec pede justo.

### Kapitel 2

Nullam dictum felis eu pede mollis pretium. Integer tincidunt.

Cras dapibus. Vivamus elementum semper nisi.

# Beispiel: <section>

```
<section>
```

```
<h1>Hauptüberschrift</h1>
```

```
<section>
```

```
<h1>Kapitel 1</h1>
```

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
```

```
<p>Nulla consequat massa quis enim. Donec pede justo.</p>
```

```
</section>
```

```
<section>
```

```
<h1>Kapitel 2</h1>
```

```
<p>Nullam dictum felis eu pede mollis pretium.</p>
```

```
<p>Cras dapibus. Vivamus elementum semper nisi.</p>
```

```
</section>
```

```
</section>
```

## Hauptüberschrift

### Kapitel 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Nulla consequat massa quis enim. Donec pede justo.

### Kapitel 2

Nullam dictum felis eu pede mollis pretium. Integer tincidunt.

Cras dapibus. Vivamus elementum semper nisi.

# Beispiel: <header>, <footer>

```
<header>  
  <h1>Willkommen</h1>  
  <p>Willkommen in der Vorlesung Webanwendungen WAW</p>  
</header>
```

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla consequat massa quis  
enim. Donec pede justo.</p>
```

```
<p>Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus  
elementum semper nisi.</p>
```

```
<footer>  
Martina Kraus  
</footer>
```

# Überschriften in HTML5

- ▶ HTML 4.01 kennt nur `<h1>` bis `<h6>` zur Strukturierung
- ▶ Gliederungsebene ergibt sich direkt aus der Überschrift

```
<h1>Überschrift 1</h1> <!-- 1. -->  
<h2>Überschrift 2</h2> <!-- 1.1 -->
```

- ▶ HTML5 hat zusätzlich `<article>` und `<section>`
- ▶ Gliederungsebene ergibt sich aus Schachtelung und Überschrift

```
<h1>Überschrift 1</h1> <!-- 1. -->  
<section>  
  <h1>Überschrift 2</h1> <!-- 1.1 -->  
</section>
```

# Beispiel: Strukturierung (klassisch)

```
<h1>Überschrift 1</h1> <!-- 1. -->  
<h2>Überschrift 2</h2> <!-- 1.1 -->  
<h3>Überschrift 3</h3> <!-- 1.1.1 -->  
<h2>Überschrift 4</h2> <!-- 1.2 -->  
<h3>Überschrift 5</h3> <!-- 1.2.1 -->
```

- 1. Überschrift 1
  - 1. Überschrift 2
    - 1. Überschrift 3
  - 2. Überschrift 4
    - 1. Überschrift 5

# Beispiel: Strukturierung (HTML5)

```
<h1>Überschrift 1</h1> <!-- 1. -->
```

```
<section>  
  <h1>Überschrift 2</h1> <!-- 1.1 -->
```

```
  <section>  
    <h1>Überschrift 3</h1> <!-- 1.1.1 -->  
  </section>  
</section>
```

```
<section>  
  <h1>Überschrift 4</h1> <!-- 1.2 -->  
  <h2>Überschrift 5</h2> <!-- 1.2.1 -->  
</section>
```

- 1. Überschrift 1
- 1. Überschrift 2
  - 1. Überschrift 3
- 2. Überschrift 4
  - 1. Überschrift 5

# Elemente mit neuer Bedeutung

- ▶ Einige Elemente haben eine neue Bedeutung
  - ▶ `<b>` - Hervorhebung von Passagen (normalerweise **fett**)
  - ▶ `<i>` - Hervorhebung von Passagen (normalerweise *kursiv*)
  - ▶ `<s>` - Hervorhebung von entfernten Inhalten
  - ▶ `<hr>` - Markiert einen inhaltlichen Bruch in einem Dokument
  - ▶ `<small>` - Markiert das „Kleingedruckte“ (Lizenzen, Copyright, Disclaimer etc.)



# Neue Elemente in HTML5

- ▶ HTML5 führt neue semantische Elemente ein
  - ▶ `<time>` - Angabe von Zeiten in maschinenlesbarer Form
  - ▶ `<mark>` - Hervorhebung von Abschnitten, die sich aus Benutzereingaben ergeben (z. B. Suchergebnisse)
  - ▶ `<figure>`, `<figcaption>` - Abbildungen o.ä. mit Beschriftung
  - ▶ `<progress>` - Fortschrittsbalken
- ▶ Zusätzlich gibt es eine Reihe von neuen Möglichkeiten bei Formularen (siehe nächster Abschnitt)

# HTML5-Formulare

# Neue Eingabe-Typen für `<input>`

- ▶ Neue Typen (*type=...*) für das `<input>`-Element
  - ▶ *search* - Suchanfragen (anderer Rahmen)
  - ▶ *tel* - Telefonnummern (keine Validierung)
  - ▶ *email* - E-Mail-Adressen (keine Validierung)
  - ▶ *url* - URL (keine Validierung)
  - ▶ *number* - Zahlen (Minimum, Maximum, Schrittweite)
  - ▶ *range* - Auswahl aus einem Bereich
  - ▶ *date*, *datetime*, *datetime-local*, *time* - Datum und Uhrzeit
  - ▶ *color* - Farben

# Beispiel: Neue Eingabe-Felder

Normales Feld: `<input type="text"><br>`

Suchfeld: `<input type="search">`

Telefon: `<input type="tel"><br>`

E-Mail: `<input type="email"><br>`

URL: `<input type="url">`

Zahlen: `<input type="number" min="0" max="24" step="0.1"><br>`

Bereiche: `<input type="range">`

Datum: `<input type="date"><br>`

Datum und Uhrzeit: `<input type="datetime"><br>`

Datum und Uhrzeit: `<input type="datetime-local"><br>`

Uhrzeit: `<input type="time">`

Farbe: `<input type="color">`

## Suchfelder

Normales Feld:

Suchfeld:

## Eingaben für Telefonnummern etc.

Telefon:

E-Mail:

URL:

## Zahlen

Zahlen:

Bereiche:

## Datum und Uhrzeit

Datum:

Datum und Uhrzeit:  UTC

Datum und Uhrzeit:

Uhrzeit:

## Farben

Farbe:

Mo	Di	Mi	Do	Fr	Sa	So
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7
Heute						

# Autovervollständigung

- ▶ Autovervollständigung des Browsers kann mit *autocomplete* kontrolliert werden
- ▶ *autocomplete="on"* - Browser darf Feld vervollständigen
- ▶ *autocomplete="off"* - Browser darf Feld nicht vervollständigen
- ▶ Vorschläge für Eingabefelder mit *<datalist>*

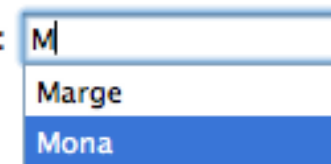
Name eines Simpsons:

```
<input type="text" autocomplete="on" list="simpsons">
```

```
<datalist id="simpsons">  
  <option value="Homer"/>  
  <option value="Marge"/>  
  <option value="Lisa"/>  
  <option value="Bart"/>  
  <option value="Lisa"/>  
  <option value="Mona"/>  
  <option value="Abe"/>  
</datalist>
```

## Datalist

Name eines Simpsons:



M

Marge

Mona

# Platzhalter

- ▶ Über *placeholder* können Platzhalter angegeben werden

```
<input type="text" name="user" placeholder="Benutzername">  
<input type="password" name="password" placeholder="Passwort">
```

## Placeholder

Benutzername	Passwort
--------------	----------

# Formularvalidierung

- ▶ HTML 4.01 benötigte JavaScript zur Validierung
- ▶ HTML5 bietet eine direkte Validierung von Formularen
  - ▶ Pflichtfelder, die ausgefüllt werden müssen
  - ▶ Syntaxregeln für bestimmte Feldtypen (z.B. E-Mail)
  - ▶ Test auf die Bedingungen eines Feldes (z.B. max und min)
  - ▶ Beliebige Syntaxregeln als Reguläre Ausdrücke
  - ▶ Beliebige Validierungen mit Hilfe von JavaScript

# Automatische Validierung

```
<form>  
  E-Mail-Adresse: <input size="30" type="email" name="adresse" id="adresse">  
  <br>  
  <input type="submit">  
</form>
```

## Automatische Validierung

E-Mail-Adresse:

Senden

! Geben Sie eine E-Mail-Adresse ein.

```
<form>  
  Vorname <input size="30" type="text" name="vorname" required><br>  
  Name <input size="30" type="text" name="nachname" required><br>  
  Telefon <input size="12" type="tel" name="telefon"><br>  
  <input type="submit">  
</form>
```

## Pflichtfelder

Vorname

Name

Telefon

Senden

! Füllen Sie dieses Feld aus.



# Validierung per REGEX

```
<form>  
  Postleitzahl: <input size="5" type="text" name="plz" pattern="[0-9]{5}"><br>  
  <input type="submit">  
</form>
```

## Validierung per REGEX

Postleitzahl:

Senden

! Ihre Eingabe muss mit dem geforderten Format übereinstimmen.