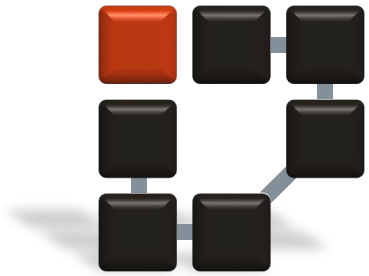
Informatik 1 für Nebenfachstudierende Grundmodul

HTML – Fortgeschrittene Techniken

Kai-Steffen Hielscher
Folienversion: 19. November 2019



Informatik 7
Rechnernetze und
Kommunikationssysteme

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Inhaltsübersicht

- Kapitel 2 - HTML
  - Einführung
  - Übersicht
  - HTML - Grundbegriffe
  - HTML - Texte und Verweise
  - **HTML - Fortgeschrittene Techniken**
  - Cascading Style Sheets CSS
  - Skripting

HTML – Fortgeschrittene Techniken

- Bereiche mit mehreren Elementen
- Tabellen
- Grafiken und Multimedia
- Formulare
- Rahmen

Bereiche mit mehreren Elementen

- Man kann mehrere HTML-Elemente wie Text, Grafiken, Tabellen usw., in einen gemeinsamen Bereich einschließen
- Dieses allgemeine Element bewirkt nichts weiter als dass es in einer neuen Zeile des Fließtextes beginnt
- Hauptzweck: Formatierung mit Hilfe von CSS
- Dieser gemeinsame Bereich wird durch das Tag-Paar `<div> ··· </div>` gekennzeichnet (div – engl. division, dt. Bereich).
- Alles, was zwischen dem Tag `<div>` und dem abschließenden Tag `</div>` steht, wird als Teil des Bereichs interpretiert
- Ein solcher Bereich und alle seine enthaltenen Elemente kann mit dem Attribut align im einleitenden div-Tag ausgerichtet werden: `align=“center“`, `align=“right“`, `align=“left“` und `align=“justify“`

Bereiche mit mehreren Elementen

- Beispiel: mehrere Absätze (Text, Grafik, Tabellen, …) gemeinsam ausrichten

```
<div align="center">
<h1>Der Mond – eine Überschrift</h1>
<img src="mond.jpg" align="middle"> Ein Bild vom Mond
<p>Ein erklärender Text zum Mond, alles wird zentriert</p>
</div>
<div align="right">
<i>&copy; 2016 by Author and Photographer</i>
</div>
```

Bereiche mit mehreren Elementen

- Zentrierter Bereich mit mehreren Elementen
  - in späteren HTML-Versionen nicht mehr unterstützt

```
<center>
<h1>Der Mond – eine Überschrift</h1>
<img src="mond.jpg" align="middle"> Ein Bild vom Mond
<p>Ein erklärender Text zum Mond, alles wird zentriert</p>
</center>
```

Bereiche mit mehreren Elementen

- weitere Möglichkeiten mit CSS und HTML 5
  - Schriftformatierung
  - Abstände, Ränder, Ausrichtung
  - Rahmen und Innenabstände
  - Hintergrundfarben und –bilder
  - Elemente positionieren
  - mehrspaltiger Textfluss

Tabellen

- Tabellen dienen der einfachen und strukturierten Darstellung der verschiedensten Informationen
- Tabellen waren oft auch das Gerüst für das Design einer Homepage (heute: Positionierung mit CSS)
  - Grundgestaltungsmittel für Seiten-Layouts des Web
  - Tabellen **mit** sichtbaren **Gitternetzlinien** (für tabellarische Daten) und
  - **ohne** sichtbare Gitternetzlinien (für mehrspaltigen Text oder für Verteilung von beliebigen Inhalten auf einer Web-Seite wie Text, Grafik oder Video)

Tabellen

- wichtige Tabellenelemente
  - `table:` Grundgerüst einer Tabelle mit `tr` und `caption`
  - `tr` (table row): Tabellenreihe mit td und th
  - `td` (table data): Tabellendaten mit verschiedenen HTML-Elementen
  - `th` (table header): Tabellenkopf zur Definition einer Kopfzelle mit versch. HTML-Elementen
  - `caption`: Tabellenüber- bzw. -unterschrift

Tabellen

- Tabellen definieren

```
<table border>
<!-- Tabelleninhalt, Tabelle mit
Gitternetzlinien -->
</table>
```

Tabellen

- Zeilen und Spalten definieren

```
<table border>
  <tr>
    <th>Kopfzelle: 1. Zeile, 1. Spalte</th>
    <th>Kopfzelle: 1. Zeile, 2. Spalte</th>
    <th>Kopfzelle: 1. Zeile, 3. Spalte</th>
  </tr>
  <tr>
    <td>Datenzelle: 2. Zeile, 1. Spalte</td>
    <td>Datenzelle: 2. Zeile, 2. Spalte</td>
    <td>Datenzelle: 2. Zeile, 3. Spalte</td>
  </tr>
  <tr>
    <td>Datenzelle: 3. Zeile, 1. Spalte</td>
    <td>Datenzelle: 3. Zeile, 2. Spalte</td>
    <td>Datenzelle: 3. Zeile, 3. Spalte</td>
  </tr>
</table>
```

Tabellen

- Spalten vordefinieren
  - automatische Formatierung durch Browser erst nach vollständigem Lesen, dauert
  - HTML 4.x

```
<table border>
  <colgroup>
    <col width=”80”>
    <col width=”100”>
    <col width=”320”>
  </colgroup>
  <tr>
    <td> Z1, S1 </td>
    <td> Z1, S2 </td>
    <td> Z1, S3 </td>
  </tr>
  <!-- weitere Zeilen -->
</table>
```

Tabellen

- Spalten vordefinieren

```
<table border>
   <colgroup width="200" span="3">
   </colgroup>
   <tr>
     <td> Z1, S1 </td>
     <td> Z1, S2 </td>
     <td> Z1, S3 </td>
   </tr>
   <!-- weitere Zeilen -->
</table>
```

Tabellen

- Spalten vordefinieren

```
<table border>
  <colgroup>
    <col width=”4*”>
    <col width=”2*”>
    <col width=”1*”>
  </colgroup>
  <tr>
    <td> Z1, S1 </td>
    <td> Z1, S2 </td>
    <td> Z1, S3 </td>
  </tr>
  <!-- weitere Zeilen -->
</table>
```

Tabellen

- Kopf, Körper und Fuß einer Tabelle definieren

```
<thead>
  <!-- Tabellenzeilen -->
</thead>
<tbody>
 <!-- Tabellenzeilen -->
</tbody>
<tfoot>
 <!-- Tabellenzeilen -->
</tfoot>
```

Grafiken und Multimedia

- Einbindung von Grafiken und Multimedia-Objekte in HTML- Dokumente – einer der interessantesten Aspekte von HTML
- Grafikformate
  - GIF, JPEG und PNG
  - alle diese Formate verfügen über spezielle Kompressionsalgorithmen: Größe der Datei kann so auch bei hoher Auflösung und Farbtiefe relativ klein gehalten werden
  - GIF: Graphics Interchange Format; Dateiendung .gif (Lizenzprobleme)
  - JPEG: Joint Photographic Experts Group; Dateiendung .jpg
  - PNG: Portable Network Graphics; Dateiendung .png

Grafiken und Multimedia

- Grafiken einbinden
  - Einbindung von Grafiken durch `img`-Element und `src`-Attribut
  - `img` ohne Gültigkeitsbereich und somit ohne Ende-Tag
  - Beispiel: `<img src="bild.jpg">`
    hier Grafik-Datei `bild.jpg` im selben Ordner wie die HTML-Datei selbst
- Alternativ-Text
  - Grafik, die eingebunden werden soll, ist nicht verfügbar oder Benutzer hat Darstellung von Grafiken unterbunden
  - Angabe eines alternativen Textes innerhalb des `<img>`-Tags mit Attribut `alt`
  - Beispiel:
    `<img src="bild.jpg" alt="Das ist eine JPEG-Grafik">`

Grafiken und Multimedia

- Breite und Höhe
  - Jede Grafik hat vordefinierte Breite und Höhe (Standard: Originalgröße)
  - Eigene Festlegung von Breite und Höhe möglich, Abänderung der Seitenverhältnisse einer Grafik
  - Breite: Attribut `width` mit Pixel-Angabe
  - Höhe: Attribut `height` mit Pixel-Angabe
  - Beispiel: `<img src="bild.jpg" width="266" height="398">`
  - auch relative Angaben zur Originalgröße mittels % möglich

Grafiken und Multimedia

- Rahmen
  - Grafik kann zusätzlich mit Rahmen versehen werden:
  - Verwendung des `<img>`-Tag mit Attribut `border` und Rahmenbreite in Pixel
  - Beispiel: `<img src=“bild.gif“ border=“3“>`
- Grafiken ausrichten
  - Ausrichtung mittels Attribut `align` und Parameter `left`, `center` und `right` am linken Rand, in der Mitte oder am rechten Rand
  - Beispiel: `<img src=“bild.gif“ align=“right“>`
- transparente Grafiken, Blind- und Fake-Grafiken, Hintergrundgestaltung, …
  - viele weitere Möglichkeiten, siehe Literatur (z.B. selfhtml)

Grafiken und Multimedia

- Grafiken als Verweis (Hyperlink)
  - Neben einfachen Text-Links auch Links mit einer Grafik möglich
  - Hauptanwendung: grafische Buttons für Navigationsleiste
  - Grafik-Datei wird im Gültigkeitsbereich des `a`-Elements notiert
  - Beispiel:

```
<a href="http://www.meineseite.de/">
<img src="button.png"></a>
```

- Grafik auch optisch als Link kennzeichnen
  - z.B. mittels Rahmen um Grafik mittels Attribut `border`:

```
<img src="button.png" border="3">
```

Grafiken und Multimedia

- Verweissensitive Grafiken (Imagemaps)
  - Imagemaps sind grafische Links, bei denen nur ausgewählte Teilbereiche der Grafik als Link fungieren
  - Mausklick auf diesen Teilbereich führt dann Verweis auf weitere Web-Seiten aus
  - Definition der einzelnen Teilbereiche mittels `map`-Element
  - Referenzierung der Imagemap mittels `name`-Attribut
  - Zugrunde liegende Grafik-Datei (auf die sich Teilbereiche beziehen) wird mittels `img`-Element eingebunden
  - Definition der einzelnen Teilbereiche der Grafik, die mit Links verbunden werden sollen, mittels `area`-Tag
  - Jeder `area`-Tag besitzt drei Attribute:
    - `href`: Angabe der Ziel-Adresse
    - `shape`: Angabe der Umrißform der Teilbereiche – `circle` (Kreis), `rect` (Rechteck), `poly` (Vieleck)
    - `coords`: Angabe der Eckkoordinaten des Umrisses

Grafiken und Multimedia

- Verweissensitive Grafiken (Imagemaps)

```
<map name="Testbild">
  <area shape=”rect” coords="1,1,249,49" href="#Anker">
  <area shape=”rect” coords="1,51,149,299" href="datei.htm">
  <area shape=”rect” coords="251,1,399,399" href="../datei.htm">
  <area shape=”rect” coords="151,51,249,299"
     href="http://www.nix.de/">
  <area shape=”rect” coords="1,301,249,399" nohref>
</map>
<img src="graphics.png" usemap="#Testbild" border=”0”>
```

Grafiken und Multimedia

- Verweissensitive Grafiken (Imagemaps): mögliche Umrissformen
  - Vieleck (`shape="rect"`): Koordinaten für x1, y1, x2, y2
    - x1: linke obere Ecke, Pixel von links
    - y1: linke obere Ecke, Pixel von oben
    - x2: rechte untere Ecke, Pixel von links
    - y2: rechte untere Ecke, Pixel von oben
  - Kreis (`shape="circle"`): Koordinaten für x, y, r
    - x: Mittelpunkt, Pixel von links
    - y: Mittelpunkt, Pixel von oben
    - r: Radius in Pixel
  - Vieleck (`shape="polygon"`): Koordinaten für x_1, y_1, x_2, y_2, …, x_n, y_n
    - x_i: Pixel der Ecke i von links
    - y_i: Pixel der Ecke i von oben
  - Koordinaten mit Grafikprogramm bestimmen

Grafiken und Multimedia

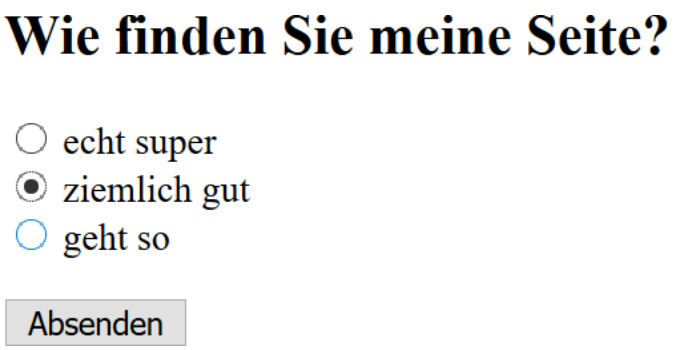
- Verweissensitive Grafiken (Imagemaps), Beispiel

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Beispiele für ImageMaps</title>
  </head>
  <body>
    <h2>Imagemap mit img und area</h2>
    <map name="meineMap">
      <area href="Dione_(Mond).htm" shape="circle"
        coords="193, 312, 125" />
      <area href="IN_saturn.html" shape="polygon" coords="260, 202,
        280, 173, 324, 158, 369, 176, 394, 226, 375, 274, 346, 290,
        321, 287, 299, 232, 300, 232" />
    </map>
    <img border="0" src="saturn.jpg" usemap="#meineMap"
      width="640" height="480" />
  </body>
</html>
```

Formulare

- Beispiel: Formular mit Radioknöpfen

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Wie finden Sie meine Seite?</title>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
</head>
<body>
<h2>Wie finden Sie meine Seite?</h2>
<form action="">
<p>
<input type="radio" name="ergebnis" value="1"> echt super<br>
<input type="radio" name="ergebnis" value="2"> ziemlich gut<br>
<input type="radio" name="ergebnis" value="3"> geht so
</p>
<input type="submit" value="Absenden">
</form>
</body>
</html>
```



Formulare

- Gute Möglichkeit, mit einem Besucher der Webseite in Kontakt zu treten
- Beispiele
  - Anmeldeformulare für Seminare, bei Behörden, Preisausschreiben,
  - Fragebögen von Umfragen
  - Formulare für Steuererklärungen
  - Einkauf bei einem Versandhandel
  - Anmeldung für einen Webservice
  - Suchanfrage, z.B. über Google
- HTML-Formulare ähnlich klassischen Papierformularen, nur mit weit mehr Möglichkeiten

Formulare

- Zweck: Sammeln von Informationen durch Interaktion mit Webseite
- Eingabefelder für
  - Texteingabe in ggf. mehrzeiligen Textfeldern
  - Auswahl aus vorgegebenen Listen
  - 1-aus-N Auswahl
  - M-aus-N Auswahl
- Verarbeitung durch Webserver: Skripting, PHP, …
- Formulare dienen zum
  - Einholen gleichartig strukturierter Auskünfte von Benutzern
  - Durchsuchen von Datenbeständen
  - Beisteuern zu Datenbeständen
  - Durchführen individueller Interaktion, z.B. Bestellung, Meinungsäußerung,
  - Suchdienste, etc.
- Achtung: Teledienste-Datenschutzgesetz (TDDSG) gebietet:
  - Nur für Formularzweck nötige Dinge abfragen (Datensparsamkeit)

Formulare

- Aufbau eines Formulars
  - Hauptelement zur Definition ist das `form`-Element
    `<form> . . . </form>`
  - form definiert das grundsätzliche Verhalten eines Formulars
  - Container für verschiedene Eingabefelder, Auswahllisten (-felder) und Schaltflächen
  - kann auch HTML-Elemente wie Tabellen, Überschriften oder Textblöcke enthalten
  - Die beiden wichtigsten Attribute des form-Elements:
    - `action` zur Definition des Ziels, an das die eingegebenen Daten gesendet werden sollen
    - im `action`-Attribut wird das Script (empfängerseitiges Programm) notiert, das die empfangenen Daten verarbeiten bzw. auswerten soll
    - Angabe einer E-Mail-Adresse (unterstützt nicht jeder Browser)
    - `method` zur Angabe der Methode, wie Daten versendet werden sollen

Formulare

- Aufbau eines Formulars
  - Zwei verschiedene Methoden von `method`
    - Methode `get` ist Standard-Methode für Formulare (d.h. Attribut `method` braucht nicht explizit gesetzt zu sein)
      - Daten des Formulars werden an die in `action` definierte Zieladresse angehängt
      - Ziel und Daten werden als ein gemeinsamer Datenstrom als Zieladresse an Browser übergeben
      - Beispiel: `http://www.ziel.de/formular.php?vorname=Fritz&nachname=Meyer&stadt=Erlangen`
      - Max. Länge 255 Zeichen gemäß URL-Adressierung
      - Daten sind in Adressleiste des Browsers sichtbar
    - Methode `post`
      - Formulardaten werden als ganzes Datenpaket an Server gesendet
      - Menge der Daten nicht auf 255 Zeichen beschränkt

Formulare

- Eingabefelder
  - Einzeilige Eingabefelder mittels HTML-Element `input` für kurze Informationen (Namen, Wohnort) mit
    - Attribut `type` und Parameter `text`
    - erzeugt im Browser einzeiliges Eingabefeld
    - Attribut `name` identifiziert Text im Eingabefeld, übertragenen Daten so eindeutig kennzeichenbar,
    - Attribut `size` und `maxlength` für Größe des Feldes und maximale Anzahl von Zeichen
    - Beispiel `<input type=“text“ name=“vorname“>`
      `<input type=“password“ name=“passw“>`
      Eingabe durch "*" maskiert
  - Mehrzeilige Eingabefelder mittels HTML-Element `textarea`
    - auch hier `name`-Attribut
    - Angabe der Größe durch Attribute `cols` und `rows`
    - Beispiel

```
<textarea name=“kommentar“ cols=“40“
  rows =“10“></textarea>
```

Formulare

- Schaltflächen
  - Schaltflächen in Formularen haben zwei Aufgaben
    - das Versenden oder das Löschen eines Formulars
    - Versenden mittels (input-) Typ `submit` und
    - Löschen mittels (input-) Typ `reset`
  - Attribut `name` für Name einer Schaltfläche und Attribut `value` für Beschriftung einer Schaltfläche
  - Beispiel

```
<input type="submit" value="Eingaben abschicken !">
<input type="reset" value="Eingaben l&ouml;schen">
```

- Neuere Variante
  - Schaltfläche `button`
  - wieder mit Attribut `type` und Parametern `submit` bzw. `reset`
  - weitere Elemente notierbar, um auf Schaltfläche zu erscheinen (z.B. Grafik)
  - Beispiel

```
<button type="submit" name="Versenden">
<img src="submit.gif"><b>Absenden</b></button>
```

Formulare

- Auswahlelemente
  - Es geht um die Auswahl aus vorgegebenen Möglichkeiten durch
    - Radiobuttons
    - Checkboxen
    - Auswahllisten
  - Radiobuttons: unter verschiedenen Möglichkeiten wird genau eine ausgewählt
    - Erstellung mit `input`-Element und `type`-Parameter `radio`
    - `value`-Attribut muss Wert enthalten
    - Beispiel

```
<input type=“radio“ name=“geschlecht“ value=“m“>
m&auml;nnlich<br>
<input type=“radio“ name=“geschlecht“ value=“w“>
weiblich<br>
<input type=“radio“ name=“geschlecht“ value=“k“>
keine Angabe<br>
```

Formulare

- Auswahlelemente
  - Checkboxen: unter verschiedenen Möglichkeiten können mehrere ausgewählt werden
    - Erstellung mit `input`-Element und `type`-Parameter `checkbox`
    - In `value` angegebene Werte werden an Empfänger übertragen und sind über in `name` angegebenen Wert identifizierbar

Formulare

- Auswahlelemente
  - Auswahllisten: Kombination aus Radiobuttons und Checkboxen
    - Benutzer kann mehrere, muss aber mindestens eine der Möglichkeiten auswählen
    - Auswahlliste mittels Elemente `select` und `option`
    - `select` definiert Auswahlliste quasi als Container für die `option`-Elemente
    - `option` legt dann die einzelnen Auswahlmöglichkeiten fest
    - Beispiel

```
<select name="Verein" size="1">
  <option>Werder Bremen</option>
  <option>Borussia Dortmund</option>
  <option>Hansa Rostock</option>
</select>
```

Formulare

- weitere Möglichkeiten
  - siehe Literatur (z.B. selfhtml)
    - Klickbuttons
    - Felder für Datei-Upload
    - versteckte Elemente
    - Gruppierung von Elementen und Label für Elemente
    - Tabulator-Reihenfolge, Tastaturkürzel und Ausgrauen
    - Buttons zum Absenden oder Abbrechen
    - Formulare mit CSS formatieren
    - Formulare verarbeiten

Rahmen

- Die Rahmen-Technologie besteht im Unterteilen des Browserfensters in mehrere kleinere Fenster, sogen. Rahmen bzw. Frames
- in den einzelnen Rahmen können dann selbst wieder HTML-Dokumente oder andere Datenquellen dargestellt werden
- Beispiel: Schema einer Webseite mit Frames

Rahmen

- Vorteile von Frames
  - übersichtliche Strukturierung
  - man kann ein (Gesamt-) Fenster in Kopffenster, Navigationsfenster und Anzeigefenster unterteilen, z.B.
    - Kopffenster mit HTML-Dokument für Logo
    - Navigationsfenster (Navigationsleiste) mit HTML- Dokument für verschiedene Links zu einzelnen Seiten der Webseite
    - Anzeigefenster zur Darstellung der jeweils gewählten Seite
  - Anwender hat immer gleiche Navigationsstruktur vor sich
  - nur Links im Navigations-Dokument muss man ändern, nicht in jedem einzelnen Dokument
- Nachteile der Verwendung von Frames
  - Höhere Ladezeiten der Webseite
  - obiges Beispiel: 4 Dokumente sind zu laden, das vierte hierbei ist das Basis-Dokument zur Erzeugung der Frames

Rahmen

- Grundaufbau
  - Definition von Frames durch HTML-Element frameset
  - wird anstelle des body-Elements notiert
  - Andere DTD, anstelle von transitional jetzt frameset
  - Grundschema:

```
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01
Frameset//EN“>
<html>
  <head>
    <title>Titel</title>
  </head>
  <frameset>
    <!-- hier folgt die Frame-Definition -->
  </frameset>
</html>
```

Rahmen

- Segmentierung der Fenster
  - Attribut `rows` bewirkt horizontale Teilung eines Fensters, notiert innerhalb des `frameset`-Tagpaar
  - Parameter sind die einzelnen Größen der Segmente
  - Reihenfolge der Größenzuweisung von oben nach unten (erster Wert: Größe des ersten Segments, …)
  - Beispiel:

```
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Frameset//EN“>
<!-- Vertikale Frames -->
<html>
  <head>
    <title>Vertikale Teilung</title>
  </head>
  <frameset rows=“50%,50%“>
    <!-- hier folgt die Frame-Definition -->
  </frameset>
</html>
```

Rahmen

- Segmentierung der Fenster
  - Attribut `cols` segmentiert das Browserfenster vertikal
  - Ebenfalls innerhalb von `<frameset>…</frameset>` notiert
  - Parameter sind die einzelnen Größen der benachbarten Segmente
  - Beispiel:

```
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Frameset//EN“>
<!-- Horizontale Frames -->
<html>
  <head>
    <title>Horizontale Teilung</title>
  </head>
  <frameset cols=“25%,75%“>
    <!-- hier folgt die Frame-Definition -->
  </frameset>
</html>
```

Rahmen

- Vollständige Frame-Definition
  - Verwendung des `frame`-Elements
  - Das `frame`-Element charakterisiert Verhalten und optische Gestaltung der einzelnen Frames
  - Festlegung, welches HTML-Dokument in welchem Fenster angezeigt wird über Attribut `src` und Pfad-Angabe des HTML-Dokuments
  - Beispiel

```
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Frameset//EN“>
<! -- Vollständiges Beispiel: frame.html -->
<html>
  <head>
    <title>Vollständiges Frame-Beispiel</title>
  </head>
  <frameset rows=“25%,75%“>
    <frame src=“oberes_frame.html“ name=“oben“>
    <frame src=“unteres_frame.html“ name=“unten“>
  </frameset>
</html>
```

Rahmen

- heute weitgehend durch CSS- und HTML5-Elemente ersetzt
- weitere Infos zu Rahmen: selfhtml