

3. Zeichen und Schrift

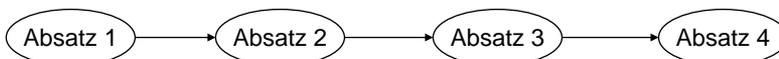
- 3.1 Medien Zeichen, Text, Schrift
- 3.2 Mikro-Typografie: Zeichensätze
- 3.3 Makro-Typografie: Gestalten mit Schrift
- 3.4 Hypertext und HTML



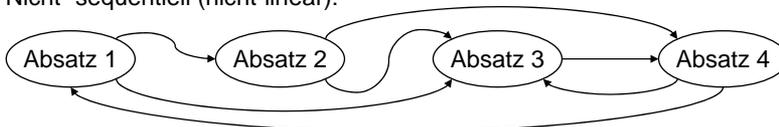
Hypertext

- *Hypertext*: Die Präsentation von Information als ein Netz von verbundenen Knoten, in dem der Leser frei, d.h. in nicht-linearer Reihenfolge navigieren kann.
- Der Begriff *hypertext* wurde von Ted Nelson geprägt, der es in seinem selbstverlegten Buch „Literary Machines“ als „nicht-sequentielles Schreiben (non-sequential writing)“ bezeichnet.

Sequentiell (linear):



Nicht- sequentiell (nicht-linear):



Information in Hypertext-Knoten

- In strengem Wortsinn: Textstück, evtl. mit Abbildungen
 - Klein genug, um eigenständige Informationseinheit zu bieten
 - Meist auf eine Seite des Anzeigergeräts passend
- In erweiterten Definitionen („Hypermedia“):
 - Klänge 
 - Filmstücke
 - Animationen 
 - ...



Hypertext-Knoten

- *Knoten*: Datenstruktur
 - erlaubt es, einen in sich geschlossenen **Information**s-Inhalt abzulegen
 - ermöglicht **Verbindungen** zu weiteren Knoten
- Andere Bezeichnungen für das Konzept des Hypertext-Knotens:
 - *frame*
 - *work space*
 - *card*
 - *lexia*
 - *web page*



Verbindung in Hypertext

- *Verbindung (link)*: Durchlaufbare Assoziation zwischen zwei **Knoten**
- *Anker*: Sichtbare Region, die mit einem Eingabegerät ausgewählt werden muss, um die Verbindung zu aktivieren
 - In den meisten Systemen dürfen sich Anker nicht überlappen.
- Detaillierungsgrad des Verbindungsziels:
 - Einfache Verbindungen: Von **Knoten** zu **Knoten**
 - Zielgenaue Verbindungen: Auswahl eines bestimmten Teils der **Information** im Ziel-Knoten



Geschichte von Hypertext

- ca. 1500, Erasmus von Rotterdam: Seitenzahlen für Querverweise in Büchern
- 1945, Vannevar Bush: Memex
- 1963, Doug Engelbart: NLS / Augment, Baumstruktur von Texten
- 1965, Ted Nelson: Xanadu
- 1975, Akscyn / McCracken (CMU): ZOG, später KMS (Knowledge Management System)
- 1976-1980, Allan Kay, Adele Goldberg, H.H. Ingalls (Xerox PARC): Objektorientierte Programmierung mit „Smalltalk“
- 1987, Bill Atkinson (Apple): HyperCard
- 1989, Tim Berners-Lee / Robert Cailleau (CERN): HTML / WWW

Hypertext im World Wide Web

- Verteilter Hypertext
 - Knoten können auf verschiedensten Rechnern weltweit liegen
- Gute Integration von Grafik, mässige Integration anderer Medientypen
- Seitenbeschreibung (HTML) orientiert an linearem Text statt Objektorientierung
 - Technologisch seit ca. 1985 überholt!
- Stark eingeschränkte Interaktivität
 - Umständliche Zusätze, z.B. Skriptsprachen
 - Kein Autorenmodus für verteilten Zugriff
- Extremer Verbreitungsgrad, extreme Informationsdichte:
 - Datenvolumen Anfang 2001 ca. 20 TeraByte (TByte = 10^{12} Byte)
 - 1999 zwei durchschnittliche WWW-Seiten nur 19 Hyperlinks voneinander „entfernt“!

WWW, HTML und HTTP

- Standardisierungsgremien:
 - IETF (Internet Engineering Task Force), z.B. HTTP
 - W3C (WWW Consortium), z.B. HTML
- Grundprinzip von HTTP:
 - Client (*Browser*) schickt Anfrage (*request*) über IP-Verbindung an Server
 - » GET: Liefere Inhalt zu URL
 - » HEAD: Wie GET, aber ohne echte Lieferung der Daten (nur „Header“)
 - » POST: Akzeptiere im Rumpf mitgelieferte Daten
 - » Diverse „Header Codes“ in der Anfrage, z.B. Browsertyp, Host, Zeichensatz-Encoding, Sprachen, ...
 - Server schickt Antwort (*response*)
 - » Hauptinhalt: HTML-Code
 - » Header-Codes auch in der Antwort

Auszeichnungssprache (Markup Language)

- Auszeichnungssprache (*markup language*) für Text
 - Text ergänzt um Angaben für die Darstellung
 - Verbreitetes Konzept; Beispiele für andere Auszeichnungssprachen: LaTeX, RTF (Rich Text Format)
- Vergleich LaTeX/HTML
 - LaTeX-Beispiel:

```
\paragraph{"Überschrift}
Text text {\it kursiver Text}
\begin{itemize}
\item Punkt in Aufz\ "ahlung
\end{itemize}
```
 - HTML-Beispiel:

```
<P>&Uuml;berschrift<BR>
Text text <I>kursiver Text</I>
<UL>
<LI>Punkt in Aufz&auml;hlung
</UL>
```

Trennung Inhalt – Darstellung

- Abstraktionsebene der Auszeichnung:
 - Entweder: „Fett 14pt“ (Mischung Inhalt-Darstellung)
 - Oder: „Überschrift Ebene 1“ (Trennung Inhalt-Darstellung) (mit separater Festlegung der Darstellung, z.B. Fett 14 pt)
- Vorteile einer starken Trennung Inhalt-Darstellung:
 - Leichtere Wartbarkeit (Regeln für die Darstellung einer Auszeichnungsklasse nur einmal definiert)
 - Bessere Plattformunabhängigkeit
 - » Konkrete optische Umsetzung („Rendering“) weitgehend der darstellenden Hardware/Software überlassen
 - Impliziter Zwang zur stilistischen Einheitlichkeit in der Darstellung
- Nachteile:
 - Verlust der Detailkontrolle über die Darstellung
 - Verlust von Flexibilität für Sonderfälle

Hypertext Markup Language HTML

- Geschichte:
 - 1969, Goldfarb, Mosher, Lorie (IBM): „Generic Markup Language“ (GML)
 - 1978, Standardisierung von GML durch ISO als „SGML“ (Standard Generic Markup Language“)
 - 1989, Tim Berners-Lee / Robert Cailleau: HTML
 - » Starke Einschränkung von SGML (spezieller Dokumententyp)
 - 1993, NCSA Mosaic Browser
 - 1999, Version 4 von HTML
 - 2000, XHTML (HTML 4.01 in XML, siehe später)
- Leistungsumfang von HTML:
 - Textattribute für die Darstellung festlegen
 - Spezielle Textformatierungen (z.B. Tabellen) definieren
 - Teile der Darstellungsfläche für interaktive Benutzereingaben vorsehen
 - Weitere Dokumente verschiedenster Art an beliebigen Stellen des Web einbinden

HTML-Syntax

`<TAG [ATTRIBUTE = "wert"]* > Inhalt [</TAG>]`

- Tags:
 - Paarweise als Beginn-/Ende-Paar `<TAGX> ... </TAGX>`
 - Einzel `z.B.
`
- Attribute:
 - Zulässige Attribute abhängig vom konkreten Tag
 - » Fast immer zulässig: CLASS, ID, LANG, STYLE
 - Attributwerte:
 - » In vielen Fällen ohne Anführungszeichen angebbbar (z.B. Zahlen)
 - » Stilistisch guter HTML-Code benutzt immer Anführungszeichen
- Zeilenumbrüche, mehrfache Leerzeichen, Tabulatoren i.A. ignoriert
- Kommentare: `<!-- ... -->`

Einfaches HTML-Beispiel

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/REC-html41/loose.dtd">
```

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Einfaches Beispieldokument HTML</TITLE>
```

```
</HEAD>
```

```
<BODY>
```

```
Ein ganz einfacher Beispieltext. <BR>
```

```
<FONT FACE="Helvetica">Fontumschaltung Helvetica</FONT> <BR>
```

```
<FONT FACE="Times">Fontumschaltung Times</FONT> <BR>
```

```
<B>Fett</B> <I>Kursiv</I>
```

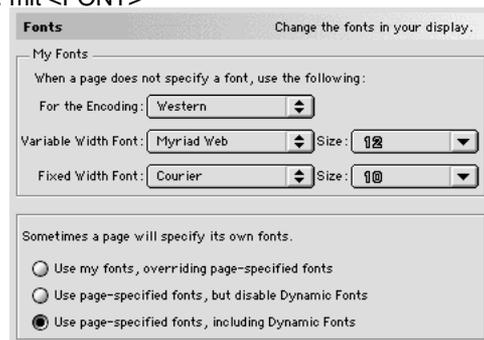
```
</BODY>
```

```
</HTML>
```

Datei: HTML1.HTML

Trennung Inhalt-Darstellung in HTML

- Starke Trennung:
 - bei Verwendung vordefinierter Textklassen
 - » z.B. <H1> für Überschriften, <ADDRESS> für Adressen
 - bei Verwendung von Cascading Style Sheets (sh. später)
- Schwache Trennung:
 - Bei expliziter Auszeichnung z.B. mit
- Prinzipiell ist in HTML keine vollständige Kontrolle über die Darstellung möglich.
 - Allerdings eine weitgehende Kontrolle für Standard-Plattformen und – Einstellungen



3. Zeichen und Schrift

- 3.1 Medien Zeichen, Text, Schrift
- 3.2 Mikro-Typografie: Zeichensätze
- 3.3 Makro-Typografie: Gestalten mit Schrift
- 3.4 Hypertext und HTML
(Fortsetzung) 

- Allgemeines 
- Textstrukturierung
- Cascading Style Sheets
- Framesets
- Medieneinbettung

Weitere Informationen: <http://selfhtml.teamone.de>

Dokumenttyp

- Verschiedene Versionen von HTML
 - Angabe benutzter Version mit DOCTYPE
 - In heutigen Browsern meist nicht überprüft!
 - Derzeit aktuelle Version: HTML 4.01 mit drei Varianten
- Strikt:
 - Modernes HTML, nicht mit alten Browsern kompatibel (vor Version 4.x)
 - Verwendung von Stylesheets und Style-Attributen

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```
- Transitional:
 - Auch ältere Konstrukte zulässig (z.B. zur Textausrichtung)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```
- Frameset:
 - Spezielle Angabe für Frameset-Dateien (sh. später)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

Kopfeinträge

- Titel
 - `<title> ... </title>`
 - Fenstertitel im Browser, Bezeichnung in Bookmarks, Anzeige bei Suchmaschinen
- Meta-Angaben für den Browser und Suchmaschinen:
 - `<meta name="author" content="Heinrich Hussmann">`
 - `<meta name="description" content="Beispieldatei ...">`
 - `<meta name="keywords" content="HTML, HEAD">`
 - `<meta name="date" content="2003-04-30">`
 - oder auch Angaben nach dem sog. "Dublin Core"-Schema:
 - `<meta name="DC.creator" content="Heinrich Hussmann">`
 - `<meta name="robots" content="noindex">` (Auslesen verbieten)
- Meta-Angaben für den Web-Server und den Browser:
 - Basis-Zeichensatz:
 - `<meta http-equiv="content-type" content="text/html" charset="ISO-8859-1">`
 - Zeitpunkt für das Löschen aus "Proxy-Servern":
 - `<meta http-equiv="expires" content="Sat, 15 Dec 2001 12:00:00 GMT">`
 - `<meta http-equiv="expires" content="43200">` (Zeit in Sekunden)

Sonderzeichen

- Zeichen können auf drei Weisen angegeben werden:
 - Direkter Zeichencode (Zeichensatz des Editors)
 - Unicode-Angabe, z.B. `®` (®), `€` (€)
 - Explizite Namen, z.B. `®`, `€`
- Wichtige Namen für deutsche Sonderzeichen:

<code>&auml;</code>	<code>&ouml;</code>	<code>&uuml;</code>	ä	ö	ü
<code>&Auml;</code>	<code>&Ouml;</code>	<code>&Uuml;</code>	Ä	Ö	Ü
<code>&szlig;</code>			ß		
- Sonderzeichen der HTML-Syntax "maskieren":

<code>&lt;</code>	<code>&gt;</code>	<code>&amp;</code>	<code>&quot;</code>	<code><</code>	<code>></code>	<code>&</code>	<code>"</code>
-----------------------	-----------------------	------------------------	-------------------------	-------------------	-------------------	--------------------	----------------

HTML-Editoren

- Software-Produkte zum bequemen Erstellen von HTML-Seiten ohne direkte Nutzung von HTML:
 - z.B. Adobe GoLive, Macromedia DreamWeaver, Microsoft FrontPage
- Vorteile:
 - Erlauben direktere Beurteilung des grafischen Effekts
 - Ersparen viele Unannehmlichkeiten von HTML
- Nachteile:
 - Gefahr der Vernachlässigung des entstehenden HTML-Codes
 - "Verunreinigen" manchmal den Code durch Editor-Artefakte
- Empfehlung:
 - Nur verwenden, wenn HTML und entstehender Code voll verstanden
 - HTML-Code-Ansicht der Werkzeuge benutzen

Elemente zur Strukturierung des Texts

- Überschriften `<h1>...</h1>` ... `<h6>...</h6>`
- Absätze `<p>...</p>`
- Unnummerierte Listen ` list item 1 `
- Nummerierte Listen ` list item 1 `
- Definitionslisten `<dl> <dt>term</dt> <dd>defn</dd> </dl>`
- Zitate `<blockquote> Zitattext </blockquote>`
- Adressen `<address> Adreßtext </address>`
- Vorformatierter Text `<pre> z.B. Programmtext </pre>`
 - Dicktengleiche (Nicht-Proportional-)Schrift
 - Umbruch und Leerzeichen wie in der HTML-Datei
- Trennlinie `<hr>`

Zeilenumbruch

- Zeilenumbruch erzwingen `
`
- Zeilenumbruch verhindern `<nobr>`
- "Geschütztes" Leerzeichen (*non-breaking space*) ` `
- Zeilenumbruch im Wort erlauben (*work break*) `<wbr>`

Tabellen (1)

- Aufteilen der Fläche in Zeilen und Spalten in flexibler Weise
 - Klassische Tabellen, Matrizen
 - Allgemeines Hilfsmittel zum Layout (bei unsichtbar gemachten Trennlinien)
 - Achtung: Tabellen werden meist erst nach vollständigem Laden angezeigt
- Allgemeine Tabellenform:

```
<table>
  <tr>
    <th>
    </th>
    <th>
    </th>
    <th>
    </th>
  </tr>
  <tr>
    <td>
    </td>
    <td>
    </td>
    <td>
    </td>
  </tr>
  <tr>
    <td>
    </td>
    <td>
    </td>
    <td>
    </td>
  </tr>
</table>
```

Tabellen (2)

- Vordefinition der Spaltenbreite (schnellere Anzeige!)
 - `<colgroup> <col width=...> ... </colgroup>`
- Unregelmässige Zellen einer Tabelle
 - Zelle über mehrere Spalten: Attribut `colspan="n"` in `<th>` und `<td>`
 - Zelle über mehrere Zeilen: Attribut `rowspan="n"` in `<th>` und `<td>`
- Rahmen
 - Attribut `border="n"` in `<table>`
- Abstände
 - Abstand Rahmen-Zellen: Attribut `cellspacing="n"` in `<table>`
 - Abstand zwischen Zellen: Attribut `cellpadding="n"` in `<table>`
- Textformatierung, Ausrichtung etc.
 - Spezielle Attribute (z.B. `align`)
 - Cascading Style Sheets (sh. unten)

Logische Auszeichnungen im Text

- Inhaltliche Beschreibung der Art des Textstücks
 - Konkrete Formatierung separat festgelegt
- Auszeichnungen:

– Betont	<code> ...</code>	<code></code>
– Stark betont	<code> ...</code>	<code></code>
– Quelltext	<code><code> ...</code>	<code></code></code>
– Beispiel	<code><samp> ...</code>	<code></samp></code>
– Tastatureingabe	<code><kbd> ...</code>	<code></kbd></code>
– Variable	<code><var> ...</code>	<code></var></code>
– Zitat	<code><cite> ...</code>	<code></cite></code>
	<code><q cite="quelle"> ...</code>	<code></q></code>
– Definition	<code><defn> ...</code>	<code></defn></code>
– Akronym	<code><acronym> ...</code>	<code></acronym></code>
– Abkürzung	<code><abbr> ...</code>	<code></abbr></code>

Physische Auszeichnungen im Text

- Beschreibung der konkreten Formatierung des Textstücks
- Auszeichnungen:
 - Fett ` ... `
 - Kursiv `<i> ... </i>`
 - Schreibmaschine `<tt> ... </tt>`
 - Unterstrichen `<u> ... </u>`
 - Durchgestrichen `<s> ... </s>`
 - Größer `<big> ... </big>`
 - Kleiner `<small> ... </small>`
 - Hochgestellt `^{...}`
 - Tiefgestellt `_{...}`

Verweise (Links)

- Klassischer Hypertext-Verweis
 - Markierter *Anker* im Text
 - *Referenz* auf andere HTML-Datei
- Syntax:
` Text `
- Beschreibung des Ziels
 - Vollständige URI (sh. nächste Folie)
 - Absolute Adressierung auf gleichem Rechner
``
 - Relative Adressierung auf gleichem Rechner
``
 - Adressierung spezieller Stellen in der Zielseite (sh. übernächste Folie)

Uniform Resource Identifier (URI)

- Offiziell: Oberbegriff von *Uniform Resource Locator (URL)* und *Uniform Resource Name (URN)*
- In der Praxis: URN kaum benutzt (obwohl hilfreiche Trennung zwischen logischer und physischer Adresse), URI = URL
- Syntax:
Protokoll : / *lokalerNetzwerkname* / *Hostname* : *Port* / *Pfad*
`http://www.mimuc.de/`
`http://Arbeitsgruppe/www.mimuc.de:8080`
`/usr/local/data/index.html`

Zielgenaue Verweise: Dokumentinterne Anker

- Hinter jeder Verweisadresse kann (mit # abgetrennt) eine Stelle in dem adressierten Dokument spezifiziert werden.
- Deklaration des Zielankers (z.B. in xyz.html):
`Text`
- Ansprechen des Zielankers:
`Text`

Stilistische Anmerkungen zu Verweisen

- Guter Stil:
 - Ankertext hat inhaltliche Bedeutung
- Beispiele:
 - Gut:
"Es steht auch vertiefende Information für Sie bereit."
 - Schlecht:
"Für vertiefende Information klicken Sie hier."

 - Gut:
"Zurück zur Institutsseite"
 - Schlecht:
"back"

3. Zeichen und Schrift

- 3.1 Medien Zeichen, Text, Schrift
- 3.2 Mikro-Typografie: Zeichensätze
- 3.3 Makro-Typografie: Gestalten mit Schrift
- 3.4 Hypertext und HTML
(Fortsetzung) 

- Allgemeines
- Textstrukturierung
- Cascading Style Sheets 
- Framesets
- Medieneinbettung

Weitere Informationen: <http://selfhtml.teamone.de>

Cascading Style Sheets (CSS)

- Vom HTML prinzipiell unabhängige Sprache zur Beschreibung von Formatierungsinformation
 - Standardisierung durch W3C
 - Besonders für HTML geeignet
- Entstehungsgeschichte:
 - Vielzahl von "Standard-Attributen" in vielen HTML-Elementen (align, pos, color, ...)
 - Vereinheitlichung in CSS (aktuelle Version 2.0)
- In HTML 4.0 wird die Ablösung "alter" Konstrukte zugunsten einheitlicher CSS-beschriebener Styles forciert.
 - Universalattribut **style**
 - Alte Schreibweise (nicht mehr empfehlenswert):
`<p>Text</p>`
 - Neue Schreibweise mit CSS-Syntax:
`<p style="font-size:250%">Text</p>`

CSS-Eigenschaften, Beispiel Schriftformatierung

- CSS-Syntax: Eigenschaft-Wert-Paare
 - Beispiel: `font-size:250%`
- Umfangreiche Liste an Eigenschaften und Maßeinheiten
- Eigenschaften zur Schriftformatierung:
 - **font** Zusammenfassung anderer Eigenschaften
 - **font-family** Gewünschte Schrift(en) mit Priorisierung
 - **font-style** Kursiv / normal
 - **font-variant** Kapitälchen (*small caps*) / normal
 - **font-size** Größe (numerisch oder ungenau)
 - **font-weight** Strichstärke (fett / mager)
 - **font-stretch** Laufweite
 - **word-spacing** Wortabstand
 - **letter-spacing** Zeichenabstand
 - **color** Farbe
 - ...

CSS-Syntax

- Eigenschaft-Wert-Paar
Eigenschaft : *Wert* z.B. `font-style:italic`
– Wenn als Wert eines HTML-Attributs: Anführungszeichen "" empfehlenswert
- Mehrere Eigenschaft-Wert-Paare
– Abtrennen mit Strichpunkt
z.B. `font-style:italic; font-size:large;`
- Anführungszeichen für Werte (z.B. bei Leerzeichen im Wert)
– Einfache Anführungszeichen ''
z.B. `font-family:'Times New Roman'`
- Mehrere Werte (Sequenz) für eine Eigenschaft
– Abtrennen mit Komma
z.B. `font-family:'Times New Roman', 'Times', serif`

Weitere CSS-Eigenschaften

- Schriftformatierung (auch mit Schriftartendatei)
- Ausrichtung und Absatzkontrolle
- Außenrand und Abstand
- Innenabstand
- Rahmen
- Hintergrundfarben und -bilder
- Listenformatierung
- Tabellenformatierung
- Pseudoformate
– z.B. `link`, `visited`, `focus`
- Positionierung und Anzeige von Elementen
- Layouts für Printmedien
- Sound-Kontrolle für Sprachausgabe
- Anzeigefenster

Einbindung von CSS in HTML (1)

- Individuell formatieren:
 - Universelles `style`-Attribut für alle HTML-Tags
 - z.B.

```
<p style="font-weight:bold; font-size:200%">
  Beispieltext</p>
```
- Zentrale Formatdefinitionen:
 - Festlegung der Style-Attribute für Standard-HTML-Elemente
 - z.B.

```
body {margin-left:100px; }
h1 { font-size:48pt;
     font-style:italic;
     border-bottom:solid thin black; }
p,li { font-size:12pt;
        line-height:14pt;
        font-family:Helvetica,Arial,sans-serif;
        letter-spacing:0.2mm;
        word-spacing:0.8mm;
        color:blue; }
```

Einbindung von CSS in HTML (2)

- Stildefinitionen im Kopfbereich der HTML-Datei

```
<style type="text/css">
... Stildefinitionen ...
</style>
```

 - Wegen Problemen älterer Browser oft Stildefinitionen als Kommentar
- Stildefinitionen in separater CSS-Datei (.css)
 - Enthält nur Stildefinitionen, kein HTML
 - Einbindung in HTML-Dateien:

```
<link rel="stylesheet" type="text/css" href=Dateireferenz>
```

Selbstdefinierte Stilklassen

- Eigene Stilklassen (außer den HTML-Elementen)
 - können frei definiert und verwandt werden
- Deklaration
 - bei der Stildefinition (mit dem Namen vorangestelltem Punkt)
 - z.B. `.navigation {font-size:16pt; color:blue;}`
- Anwendung
 - mit dem universellen `class`-Attribut aller HTML-Tags
 - z.B. `<li class=navigation>Home`

Blockweise Formatierung mit CSS

- Ganze Textbereiche einheitlich formatieren
- Verwendung des *Inline-Elements* ` ... `
 - Keinerlei Effekt auf die Dokumentstruktur
 - Kann Text oder andere Inline-Elemente enthalten
 - Völlig äquivalent zur Wiederholung der angegebenen Stilangaben bei allen enthaltenen HTML-Elementen (mit `style`)
- Verwendung des *allgemeinen Blockelements* `<div> ... </div>`
 - Kann Text oder andere Blockelemente enthalten, z.B. auch Grafiken
 - Weitergabe der angegebenen Stilangaben zu allen enthaltenen HTML-Elementen
 - Kann mit der CSS-Eigenschaft `position` absolut positioniert werden
 - Kann mit Skripten ein- und ausgeblendet werden
 - Anmerkung: Oft benannt nach dem alten Netscape-spezifischen Element "layer"

3. Zeichen und Schrift

- 3.1 Medien Zeichen, Text, Schrift
- 3.2 Mikro-Typografie: Zeichensätze
- 3.3 Makro-Typografie: Gestalten mit Schrift
- 3.4 Hypertext und HTML
(Fortsetzung) 

- Allgemeines
- Textstrukturierung
- Cascading Style Sheets
- Framesets 
- Medieneinbettung

Weitere Informationen: <http://selfhtml.teamone.de>

Framesets

- Einteilung einer Seite in separate Segmente (*frames*)
 - Die Gesamtseite definiert ein sogenanntes *frameset*.
 - Jedes Einzelframe liegt in einer Einzeldatei.
 - Anzeige der Frames ist unabhängig (werden separat geladen).
- Grundgerüst eines Framesets:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<title>Text des Titels</title>
</head>
<frameset ...> <!-- Frameset-Definition -->
  <frame ...> <!-- Framefenster-Definition -->
</frameset>
</html>
```

Wird angezeigt, wenn der Browser keine Frames anzeigen kann

Anzeige von Inhalten in Framesets

- Bei Verweisen kann mit dem **target**-Attribut festgelegt werden, in welchem Frame die Anzeige erfolgt.
 - **target="_blank"** bedeutet Anzeige in einem neuen Fenster

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Text des Titels</title>
</head>

<h1>Navigation</h1>
<p>
<a href="allgem.html" target="Daten"><b>Allgemeines</b></a><br>
<a href="styles.html" target="Daten"><b>Styles</b></a><br>
<a href="table.html" target="Daten"><b>Tabellen</b></a>
</p>

</body>
</html>
```

Vor- und Nachteile von Framesets

- Vorteile:
 - Elegante Gestaltungsmöglichkeiten
 - Navigationshilfen bleiben auch beim Blättern in angezeigter Information am gleichen Platz
 - Nachladen von Einzelinformation u.U. schneller als ohne Frames
 - Parallele Anzeige von Information z.B. zu Vergleichszwecken
- Nachteile:
 - Inkompatibilität mit älteren Browsern
 - Gesamtladezeit schlechter als ohne Frames
 - Einzelansichten nicht mehr als Ganzes adressierbar
 - » Bruch mit den Grundparadigmen von HTML?
- Empfehlungen:
 - Frames nur da einsetzen, wo wirklich sinnvoll!
 - Idealerweise Frame-freie Alternative (nicht nur Fehlertext) anbieten

3. Zeichen und Schrift

- 3.1 Medien Zeichen, Text, Schrift
- 3.2 Mikro-Typografie: Zeichensätze
- 3.3 Makro-Typografie: Gestalten mit Schrift
- 3.4 Hypertext und HTML
(Fortsetzung) 

- Allgemeines
- Textstrukturierung
- Cascading Style Sheets
- Framesets
- Medieneinbettung 

Weitere Informationen: <http://selfhtml.teamone.de>

Integration von Bildern

- Bilder einbinden mit ``
- Attribut `src` gibt Quelle an (auch von anderen Servern möglich)
 - Achtung Copyright-Fragen!
- Größenangaben mit `width` und `height`
 - Bei Angabe beider Werte Verzerrung möglich
- Bilder können auch als Inhalt eines Verweises vorkommen
 - z.B. grafische Navigationsleisten

```
<html>
...
<body>

<h1>Ein JPEG-Bild des Eiffelturms</h1>

<p></p>

</body>
</html>
```

Integration anderer Dateien

- Prinzipiell alle Dateien einbettbar
 - mit dem `<object>`-Tag
 - als Hyperlinks
- Beispiel zu Sound siehe unten
 - Achtung nur in neuesten Browsern unterstützt
 - Ältere Variante: `<embed>`-Tag

```
<p>Sound-Objekt
  <object data="../sounds/bgndmusic.mid" type="audio/midi">
  Ihr Browser kann das Objekt leider nicht anzeigen!
</object>
</p>
```

```
<p>Sound als Link<br>
  <a href="../sounds/technobop.mid"
  type="audio/midi">Bitte klicken!</a>
</p>
```

MIME

- MIME = Multipurpose Internet Mail Extensions
 - In HTML mit dem `type`-Attribut an vielen Stellen angebbar (z.B. `<link>`, `<object>`)
 - Erleichtert dem Browser (bzw. seinem Benutzer) die Entscheidung, wie Dateien zu behandeln sind
 - Jeder Browser führt eine Liste der akzeptierten MIME-Extensions und Regeln für die Behandlung (z.B. speichern, Programm aufrufen)
 - Liste siehe <http://www.iana.org/assignments/media-types>
- Syntax:
 - Medientyp / Untertyp*
 - Medientypen: text, image, video, audio, application, ...
 - Subtypen, die auf dem Server auszuführen sind, beginnen meist mit x-
 - Hersteller- (*vendor*-)spezifische Subtypen im speziellen Unterbaum "vnd."