

# 11.Vorlesung Grundlagen der Informatik

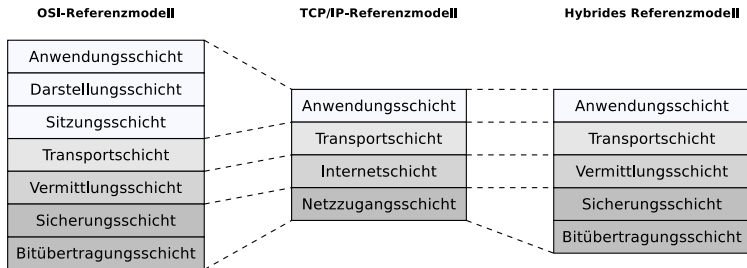
Dr. Christian Baun

Hochschule Darmstadt  
Fachbereich Informatik  
christian.baun@h-da.de

22.12.2011

# Wiederholung vom letzten Mal

- Vermittlungsschicht
  - Adressierung (IP-Adressen)
- Transportschicht
  - Ports und Portnummern, Sockets
  - Transportprotokolle (UDP, TCP)
- Sitzungsschicht
- Darstellungsschicht
- Anwendungsschicht

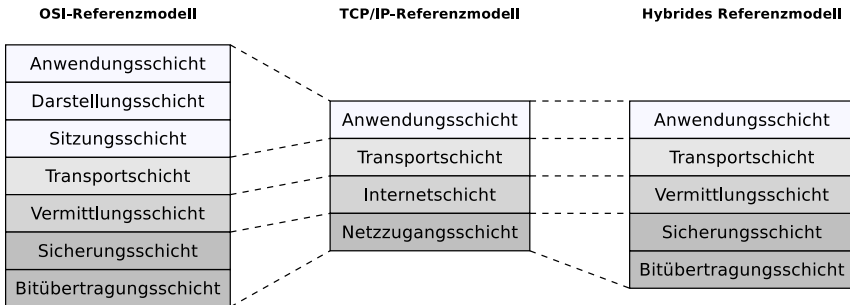


# Heute

- Anwendungsschicht
  - Anwendungsprotokolle
    - Fernsteuerung von Computern mit Telnet
    - Übertragung von Daten mit dem Hypertext Transfer Protocol (HTTP)
- Informationen in Hypertext-Systemen mit Auszeichnungssprachen darstellen
  - Hypertext Markup Language (HTML)
  - Inhalt und Layout der Informationen voneinander trennen
    - Extensible Markup Language (XML)
    - Transformation von XML-Dokumenten mit XSLT

# Anwendungsschicht

- Enthält Anwendungsprotokolle und darauf aufbauende Dienste u.a. zur Datenübertragung, Synchronisierung und Fernsteuerung von Rechnern und Namensauflösung



- Geräte in der Anwendungsschicht: keine
- Protokolle in der Anwendungsschicht: DNS, NTP, SSH, HTTP, FTP...

# Telnet (Telecommunication Network)

- Protokoll zur Fernsteuerung von Rechnern
- 1971 im Rahmen des ARPANET-Projekts erfunden
- Bietet zeichenorientierten Datenaustausch über eine TCP-Verbindung
- Verwendet standardmäßig Port 23
- Software, die das Protokoll implementiert, heißt auch einfach Telnet
  - Besteht aus den beiden Diensten Telnet-Client und Telnet-Server
- Eignet sich nur für Anwendungen ohne grafische Benutzeroberfläche
- Wird heute häufig zur Fehlersuche bei anderen Diensten und zur Administration von Datenbanken eingesetzt
- Nachteil: Keine Verschlüsselung!
  - Auch Passwörter werden im Klartext versendet
- Nachfolger: Secure Shell Protokoll (SSH)

# Telnet und das virtuelle Netzwerkterminal

- Telnet basiert auf dem Standard NVT
  - NVT (Network Virtual Terminal) = virtuelles Netzwerkterminal
    - Konvertierungskonzept für unterschiedliche Codes und Datenformate
    - Herstellerunabhängige Schnittstelle
    - Wird von allen Telnet-Implementierungen unterstützt
    - Verwendet auch andere bekannte Protokolle wie FTP und SMTP
    - Telnet-Clients konvertieren die Tasteneingaben und Kontrollanweisungen in das NVT-Format und übertragen diese Daten an den Telnet-Server
- NVT arbeitet mit Informationseinheiten von je 8 Bit = 1 Byte
- Telnet verwendet ASCII-Zeichen mit je 7 Bit
- Das höchstwertige Bit wird mit Null aufgefüllt, um auf 8 Bit zu kommen

# Kontrollanweisungen in Telnet

- Die Tabelle enthält die Kontrollanweisungen von NVT
  - Die ersten 3 Kontrollzeichen versteht jeder Telnet-Client und -Server
  - Die übrigen 5 Kontrollzeichen sind optional

Name	Code	Dezimalwert	Beschreibung
NULL	NUL	0	No operation
Line Feed	LF	10	Zeilenvorschub (nächste Zeile, gleiche Spalte)
Carriage Return	CR	13	Wagenrücklauf (gleiche Zeile, erste Spalte)
BELL	BEL	7	Hörbares oder sichtbares Signal auf dem Display
Back Space	BS	8	Cursor eine Position zurück bewegen
Horizontal Tab	HT	9	Cursor zum nächsten horizontalen Tabulatorstopp bewegen
Vertical Tab	VT	11	Cursor zum nächsten vertikalen Tabulatorstopp bewegen
Form Feed	FF	12	Cursor in die erste Spalte der ersten Zeile bewegen und Terminal löschen

- Ein Zeilenende im Text wird als Wagenrücklauf – Carriage Return (CR), gefolgt von einem Zeilenvorschub – Line Feed (LF) übertragen
- Ein Wagenrücklauf im Text wird als CR, gefolgt von einem NUL-Zeichen (alle Bits 0) übertragen

# Hypertext-Übertragungsprotokoll

- Das Hypertext Transfer Protocol (HTTP) ist ein zustandsloses Protokoll zur Übertragung von Daten
- Ab 1989 von Roy Fielding, Tim Berners-Lee und anderen am CERN entwickelt
- Ist gemeinsam mit den Konzepten URL und HTML die Grundlage des World Wide Web (WWW)
- Haupteinsatzzweck: Webseiten aus dem World Wide Web (WWW) in einen Webbrowser laden
- Zur Kommunikation ist HTTP auf ein zuverlässiges Transportprotokoll angewiesen
  - In den allermeisten Fällen wird TCP verwendet
- Jede HTTP-Nachricht besteht aus:
  - Nachrichtenkopf (*HTTP-Header*): Enthält u.a. Informationen zu Kodierung, zur gewünschten Sprache, über den Browser und Inhaltstyp
  - Nachrichtenkörper (*Body*): Enthält die Nutzdaten



# HTTP-Anfragen (1/2)

- Wird über HTTP auf eine URL (z.B. `http://www.informatik.hs-mannheim.de/~baun/index.html`) zugegriffen, wird an den Rechner mit dem Hostnamen `www.informatik.hs-mannheim.de` die Anfrage gesendet, die Ressourcen `/~baun/index.html` zurückzusenden
- Zuerst wird der Hostname via DNS in eine IP-Adresse umgewandelt
- Über TCP wird zu Port 80 (auf dem der Web-Server üblicherweise arbeitet) folgende HTTP-GET-Anforderung gesendet

```
GET /~baun/index.html HTTP/1.1
Host: www.informatik.hs-mannheim.de
User-Agent: Mozilla/5.0 (X11; U; Linux i686; de; rv:1.9.2.18) Gecko/20110628 Ubuntu/10.10 (
  maverick) Firefox/3.6.18
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de-de;de;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
...
```

# HTTP-Anfragen (2/2)

- So ein großer Nachrichtenkopf ist eigentlich nicht nötig
- Die folgende HTTP-GET-Anforderung genügt völlig

```
GET /~baun/index.html HTTP/1.1  
Host: www.informatik.hs-mannheim.de
```

- Der Nachrichtenkopf wird mit einer Leerzeile (bzw. 2 aufeinanderfolgenden Zeilenenden) vom Nachrichtenkörper abgegrenzt
- In diesem Fall hat die HTTP-Anforderung keinen Nachrichtenkörper

# HTTP-Antworten (1/2)

- Die HTTP-Antwort des Web-Servers besteht aus einem Nachrichtenkopf und dem Nachrichtenkörper mit der eigentlichen Nachricht
  - In diesem Fall enthält der Nachrichtenkörper den Inhalt der Datei `index.html`

```
HTTP/1.1 200 OK
Date: Sun, 04 Sep 2011 15:19:13 GMT
Server: Apache/2.2.17 (Fedora)
Last-Modified: Mon, 22 Aug 2011 12:37:04 GMT
ETag: "101ec1-2157-4ab17561a3c00"
Accept-Ranges: bytes
Content-Length: 8535
Keep-Alive: timeout=13, max=499
Connection: Keep-Alive
Content-Type: text/html

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

# HTTP-Antworten (2/2)

- Jede HTTP-Antwort enthält einen **HTTP-Statuscode** (3 Ziffern) und eine Textkette, die den Grund für die Antwort beschreibt

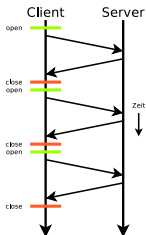
Statuscode	Bedeutung	Beschreibung
1xx	<b>Informationen</b>	Anfrage erhalten, Prozess wird fortgeführt
2xx	<b>Erfolgreiche Operation</b>	Aktion erfolgreich empfangen. Antwort wird an den Client gesendet
3xx	<b>Umleitung</b>	Weitere Aktion des Clients erforderlich
4xx	<b>Client-Fehler</b>	Anfrage des Clients fehlerhaft
5xx	<b>Server-Fehler</b>	Fehler, dessen Ursache beim Server liegt

- Die Tabelle enthält einige populäre HTTP-Statuscodes

Statuscode	Beschreibung	Beschreibung
118	Connection timed out	Zeitüberschreitung beim Ladevorgang
200	OK	Anfrage erfolgreich bearbeitet. Ergebnis wird in der Antwort übertragen
202	Accepted	Anfrage akzeptiert, wird aber zu einem späteren Zeitpunkt ausgeführt
204	No Content	Anfrage erfolgreich durchgeführt. Antwort enthält bewusst keine Daten
301	Moved Permanently	Ressource verschoben. Die alte Adresse ist nicht länger gültig
307	Temporary Redirect	Ressource verschoben. Die alte Adresse bleibt gültig
400	Bad Request	Anfrage-Nachricht war fehlerhaft aufgebaut
401	Unauthorized	Anfrage kann nicht ohne gültige Authentifizierung durchgeführt werden
403	Forbidden	Anfrage mangels Berechtigung des Clients nicht durchgeführt
404	Not Found	Ressource vom Server nicht gefunden
500	Bad Request	Unerwarteter Serverfehler

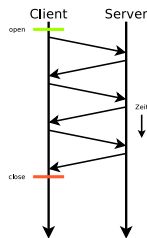
# Das Protokoll HTTP

- Es existieren 2 Protokollversionen: HTTP/1.0 und HTTP/1.1



- HTTP/1.0: Vor jeder Anfrage wird eine neue TCP-Verbindung aufgebaut und nach Übertragung der Antwort standardmäßig vom Server wieder geschlossen
  - Enthält ein HTML-Dokument z.B. 10 Bilder, sind 11 TCP-Verbindungen nötig

- HTTP/1.1: Ein Client kann durch den Headereintrag `Connection: keep-alive` anweisen, dass kein Verbindungsabbau durchgeführt wird
  - So kann die Verbindung immer wieder verwendet werden
  - Für das HTML-Dokument mit 10 Bildern ist so nur 1 TCP-Verbindung nötig
    - Dadurch kann die Seite schneller geladen werden



# HTTP-Methoden

- Das HTTP-Protokoll enthält einige Methoden für Anfragen

HTTP	Beschreibung
PUT	Neue Ressource auf den Web-Server hochladen
GET	Ressource vom Web-Server anfordern
POST	Daten zum Web-Server hochladen, um Ressourcen zu erzeugen
DELETE	Eine Ressource auf dem Web-Server löschen
HEAD	Header einer Ressource vom Web-Server anfordern, aber nicht den Body
TRACE	Liefert die Anfrage so zurück, wie der Web-Server sie empfangen hat. Hilfreich für die Fehlersuche
OPTIONS	Liste der vom Web-Server unterstützten HTTP-Methoden anfordern
CONNECT	SSL-Tunnel mit einem Proxy herstellen

HTTP ist ein zustandsloses Protokoll. Über Cookies in den Header-Informationen sind dennoch Anwendungen realisierbar, die Status- bzw. Sitzungseigenschaften erfordern weil sie Benutzerinformationen oder Warenkörbe den Clients zuordnen.

# Eine Möglichkeit, Web-Server zu testen, ist telnet (1/2)

```
$ telnet www.informatik.hs-mannheim.de 80
Trying 141.19.145.2...
Connected to anja.ki.fh-mannheim.de.
Escape character is '^]'.
GET /~baun/index.html HTTP/1.0

HTTP/1.1 200 OK
Date: Sun, 04 Sep 2011 21:43:53 GMT
Server: Apache/2.2.17 (Fedora)
Last-Modified: Mon, 22 Aug 2011 12:37:04 GMT
ETag: "101ec1-2157-4ab17561a3c00"
Accept-Ranges: bytes
Content-Length: 8535
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
...
</body>
</html>
Connection closed by foreign host.
```

# Eine Möglichkeit, Web-Server zu testen, ist telnet (2/2)

```
$ telnet www.informatik.hs-mannheim.de 80
Trying 141.19.145.2...
Connected to anja.ki.fh-mannheim.de.
Escape character is '^]'.
GET /~baun/test.html HTTP/1.0
```

```
HTTP/1.1 404 Not Found
Date: Sun, 04 Sep 2011 21:47:26 GMT
Server: Apache/2.2.17 (Fedora)
Content-Length: 301
Connection: close
Content-Type: text/html; charset=iso-8859-1
```

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /~baun/test.html was not found on this server.</p>
<hr>
<address>Apache/2.2.17 (Fedora) Server at anja.ki.hs-mannheim.de Port 80</address>
</body></html>
Connection closed by foreign host.
```



## Not Found

The requested URL /~baun/test.html was not found on this server.

Apache/2.2.17 (Fedora) Server at www.informatik.hs-mannheim.de Port 80



# Hypertext- und Hypermedia-Systeme

- Surfen ist anders als Blättern in einem Buch
- Webseiten enthalten Verweise (Hyperlinks, Links) auf andere Webseiten
- Ein System vernetzter Webseiten nennt man **Hypertext-System**
- Hypertext wird in **Auszeichnungssprachen** geschrieben
  - Auszeichnungssprachen beinhalten neben Format-Anweisungen auch Befehle für Hyperlinks
  - Die bekannteste Auszeichnungssprache ist die Hypertext Markup Language (HTML)
- Bilder, Audio-Sequenzen, Video-Clips sind weitere mögliche Medien auf einer Webseite
  - **Hypermedia** bezeichnet eine Variante von Hypertext unter expliziter Betonung des multimedialen Aspektes
- Browser sind zur Anzeige von Webseiten und zur Navigation nötig

# Uniform Resource Locator (URL)

- Lokalisation und Identifizierung von Ressourcen (Webseiten) im Internet
- Eingabe der Web-Adresse in die Adresszeile des Browsers
- Web-Adressen sind nach einem bestimmten Muster (**URL**) aufgebaut:  
Protokoll://Rechneradresse/Dokumentenadresse
- Via Browser fordert man eine Webseite von einem adressierten Rechnern an
- **Web-Server** bezeichnet die Software, die für die Auslieferung der Webseite zuständig ist, sowie einen Rechner (Server), auf dem der Web-Server (Software) läuft

# Hypertext Markup Language (HTML)

- Ermöglicht die Festlegung von Aufbau und Struktur von Webseiten
- November 1992: Erste Version der HTML-Spezifikation am CERN
- Text erhält durch Auszeichnungen (englisch: *markup*) von Textteilen eine Struktur
- Browser übersetzt den Quelltext und stellt den Inhalt formatiert dar
- HTML-Elemente (**Tags**)
  - Zeichenfolgen, die mit spitzen Klammern anfangen und enden
  - Sind in HTML genau festgelegt
  - Ein schließendes Tag folgt auf ein öffnendes Tag
    - Das schließende Tag enthält zusätzlich einen Slash (/)
  - Korrekte Schachtelung ist wichtig

Sehr gute deutschsprachige Quelle um HTML zu lernen und nachzuschlagen

SELFHTML – <http://de.selfhtml.org>

# Gerüst einer HTML-Seite

- Ein HTML-Dokument besteht aus 3 Bereichen:
  - ① Dokumenttypdeklaration (Doctype)
    - Gibt die verwendete Dokumenttypdefinition (DTD) an
    - Legt die Reihenfolge und Verschachtelung der Elemente und die Inhalte von Attributen fest
  - ② HTML-Kopf (HEAD)
    - Enthält Informationen zum Dokument, die üblicherweise nicht im Anzeigebereich des Browsers dargestellt werden
  - ③ HTML-Körper (BODY)
    - Enthält Informationen, die üblicherweise im Anzeigebereich des Browsers zu sehen sind

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Titel der Webseite</title>
    <!-- Weitere Kopfinformationen -->
  </head>
  <body>
    <p>Inhalt der Webseite</p>
  </body>
</html>
```

# Darstellung des Textes beeinflussen

- Texte werden mit `<b>` (*bold*) in Fettschrift angezeigt

```
<b>Dieser Text wird in Fettschrift ausgegeben.</b>
```

- Texte werden mit `<i>` (*italic*) kursiv angezeigt

```
<i>Dieser Text wird kursiv ausgegeben</i> und <b><i>dieser fett und kursiv</i></b>.
```

- Texte erhalten mit `<tt>` (*teletyper*) eine Schrift mit fester Zeichenbreite (⇒ nichtproportionale Schriftart)

```
<tt>Dieser Text wird in nicht proportialer Schrift ausgegeben.</tt>
```

- Texte werden mit `<u>` unterstrichen und mit `<s>` durchgestrichen

```
<u>Dieser Text ist unterstrichen</u> und <s>dieser ist durchgestrichen.</s>
```

- Texte werden mit `<sup>` hochgestellt und mit `<sub>` tiefgestellt

```
Man kann Texte <sup>hochstellen</sup> und <sub>tiefstellen</sub>.
```

## Sonderzeichen

- Verwendet man Sonderzeichen (z.B. Umlaute), kann man diese direkt eingeben und im Kopf die Kodierung angeben

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

- Alternativ kann man Sonderzeichen und Umlaute direkt mit **benannten Zeichen** eingeben

Zeichen	HTML-Code	Zeichen	HTML-Code	Zeichen	HTML-Code
ä	&auml;	"	&quot;	←	&larr;
Ä	&Auml;	<	&lt;	→	&rarr;
ö	&ouml;	>	&gt;	⇐	&lArr;
Ö	&Ouml;	&	&amp;	⇒	&rArr;
ü	&uuml;	\$	&sect;	–	&ndash;
Ü	&Uuml;	©	&copy;	—	&mdash;
ß	&szlig;	®	&reg;	∞	&infin;
€	&euro;	Leerzeichen	&nbsp;	≠	&ne;

# Überschriften

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>&Uuml;berschriften definieren</title>
</head>
<body>
<h1>&Uuml;berschrift 1. Ordnung</h1>
<h2>&Uuml;berschrift 2. Ordnung</h2>
<h3>&Uuml;berschrift 3. Ordnung</h3>
<h4>&Uuml;berschrift 4. Ordnung</h4>
<h5>&Uuml;berschrift 5. Ordnung</h5>
<h6>&Uuml;berschrift 6. Ordnung</h6>
</body>
</html>
```

## Überschrift 1. Ordnung

### Überschrift 2. Ordnung

#### Überschrift 3. Ordnung

##### Überschrift 4. Ordnung

###### Überschrift 5. Ordnung

###### Überschrift 6. Ordnung

- **h** = *heading* (Überschrift)

# Textabsätze

- Textabsätze leitet `<p>` (*paragraph* = Absatz) ein und `</p>` beendet sie

```
<p>Hier beginnt ein Absatz, und hier ist er zu Ende.</p>
```

- Textabsätze kann man ausrichten

```
<p align="center">Zentrierter Absatz.</p>
<p align="right">Rechtsbündig ausgerichteteter Absatz.</p>
<p align="left">Linksbündig ausgerichteteter Absatz.</p>
<p align="justify">Absatz, der im Blocksatz ausgerichtet ist.</p>
```

- Zeilenumbrüche erzwingt man mit `<br>` (*break* = Umbruch)

```
<p>Hier ist nun der Zeilenumbruch.<br>Hier geht die neue Zeile weiter.</p>
```



# Unnummerierte Listen

- Mit `<ul>` (*unordered list*) erstellt man eine unnummerierte Liste (*Bulletliste*):
- Mit `<li>` (*list item*) erstellt man einen Listenpunkt:

- Erstens
- Zweitens
- Drittens
  
- Erstens
- Zweitens
- Drittens
  
- Erstens
- Zweitens
- Drittens

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Unnummerierte Listen</title>
</head>
<body>

<ul>
  <li>Erstens</li>
  <li>Zweitens</li>
  <li>Drittens</li>
</ul>

<ul type="circle">
  <li>Erstens</li>
  <li>Zweitens</li>
  <li>Drittens</li>
</ul>

<ul type="square">
  <li>Erstens</li>
  <li>Zweitens</li>
  <li>Drittens</li>
</ul>

</body>
</html>
```

# Nummerierte Listen

- Mit `<ol>` (*ordered list*) erstellt man eine nummerierte Liste (*Bulletliste*):
- Mit `<li>` (*list item*) erstellt man einen Listenpunkt:

1. Erstens
2. Zweitens
3. Drittens

- a. Erstens
- b. Zweitens
- c. Drittens

- I. Erstens
- II. Zweitens
- III. Drittens

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Unnummerierte Listen</title>
</head>
<body>

<ol>
  <li>Erstens</li>
  <li>Zweitens</li>
  <li>Drittens</li>
</ol>

<ol type="a">
  <li>Erstens</li>
  <li>Zweitens</li>
  <li>Drittens</li>
</ol>

<ol type="I">
  <li>Erstens</li>
  <li>Zweitens</li>
  <li>Drittens</li>
</ol>

</body>
</html>

```

# Verweise (Links)

- Mit Verweisen strukturiert man große Dokumente und Inhalte
- Durch Verweise auf andere Webseiten wird aus dem einfachen Text eine Hypertextstruktur
- Plant man z.B. eine Einstiegsseite und verschiedene Dateien für Unterseiten, braucht man in der Einstiegsseite Verweise zu allen Unterseiten und in jeder Unterseite einen Verweis zur Einstiegsseite
  - Dadurch wird aus der losen Dateisammlung eine zusammenhängende Homepage
- Verweise realisiert man mit dem Element `a` (*anchor* = Anker)

```
<a href="http://www.h-da.de">Hochschule Darmstadt</a><br>  
<a href="/">Zur&uuml;ck zur Startseite</a><br>  
<a href="..">Eine Ebene zur&uuml;ck</a><br>
```

- Das Attribut `href` (*hyper reference*) enthält das Verweisziel
- Zwischen den Tags steht der Text, der als Verweise angeboten wird und vom Benutzer angeklickt werden kann

# Absolute und Relative Verweise (1/2)

- **Absolute Verweise** verweisen immer auf das gleiche Ziel
- Beispiele:
  - `http://hostname`  
Startseite eines Webserver mit dem Hostnamen `hostname`
  - `http://hostname:8080`  
Startseite eines Webserver mit dem Hostnamen `hostname` unter der Portnummer 8080
  - `http://hostname/verzeichnis/`  
Startseite eines im Verzeichnis `verzeichnis` eines Webserver mit dem Hostnamen `hostname`
  - `http://hostname/verzeichnis/dateiname`  
Datei `dateiname` im Verzeichnis `verzeichnis` eines Webserver mit dem Hostnamen `hostname`
  - `file:///verzeichnis/dateiname`  
Datei `dateiname` im Verzeichnis `verzeichnis` auf dem lokalen Rechner

# Absolute und Relative Verweise (1/2)

- **Relative Verweise** verweisen auf ein Ziel dessen Lage relativ zur Adresse der HTML-Datei ist, die den Verweis enthält
- Nützlich u.a. wenn Dateien in der Gruppe beweglich sein sollen
- Szenario:
  - Webseiten auf einen anderen Webserver verschieben
  - Webseiten in ein anderes Verzeichnis auf dem Webserver verschieben
  - Webseiten auf CD-ROM weitergeben
- Beispiele:
  - `dateiname`  
Datei `dateiname` im gleichen Verzeichnis auf dem gleichen Webserver
  - `verzeichnis/dateiname`  
Datei im Verzeichnis `verzeichnis` auf dem Webserver (relativer Pfad)
  - `../dateiname`  
Datei im Elternverzeichnis auf dem Webserver (relativer Pfad)
  - `/verzeichnis/dateiname`  
Datei im Verzeichnis `verzeichnis` auf dem gleichen Webserver (absoluter Pfad)

# Bilder einbinden

- Bilder bindet man mit dem Tag `img` (*image*) ein

```

```

- Das Image-Tag umschließt keinen Inhalt, deswegen fehlt ein schließendes Tag
- Attribut `src` enthält den Dateinamen des Bildes
- Eventuelle Unterordner oder URLs müssen hier angegeben werden
- Attribut `alt` enthält einen Alternativtext, für den Fall, dass die Grafik nicht angezeigt werden kann
  - Der Alternativtext kann ein Wort oder eine Kurzbeschreibung sein
- Mit den Attributen `width` und `height` kann man die Breite und Höhe in Pixeln oder als Prozentangabe festlegen

```

```

```

```

# Tabellen

- Mit Tabellen kann man tabellarische Daten darstellen oder das Layout von Webseiten beeinflussen
  - Optisch unterscheidet man zwischen **Tabellen mit Gitternetzlinien** (für tabellarische Daten), und **Blinden Tabellen** ohne sichtbare Gitternetzlinien für die Verteilung von Inhalten auf einer Webseite
- Der Tag `<table>` leitet eine Tabelle ein
  - Soll die Tabelle sichtbare Gitternetzlinien haben, muss das Attribut `border` im einleitenden Tag `<table>` einen Wert größer 0 haben
    - Der angegebene Wert ist die Breite des Rahmens in Pixeln
  - Der Tag `<tr>` (*table row*) leitet eine neue Tabellenzeile ein
  - Eine Tabelle kann Kopfzellen und gewöhnliche Datenzellen enthalten
    - Text in Kopfzellen wird fett ausgegeben und zentriert ausgerichtet
  - Der Tag `<th>` (*table header*) leitet eine Kopfzelle ein
  - Der Tag `<td>` (*table data*) leitet eine normale Datenzelle ein
- In einer Tabellenzelle können u.a. Texte, Bilder oder sogar eine weitere Tabelle stehen

# Tabelle mit Gitternetzlinien

Zelle 1/1	Zelle 2/1	Zelle 3/1
Zelle 1/2	Zelle 2/2	Zelle 3/2
Zelle 1/3	Zelle 2/3	Zelle 3/3

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Tabelle mit Gitternetzlinien</title>
</head>
<body>

<table border="1">
  <tr>
    <td>Zelle 1/1</td>
    <td>Zelle 2/1</td>
    <td>Zelle 3/1</td>
  </tr>
  <tr>
    <td>Zelle 1/2</td>
    <td>Zelle 2/2</td>
    <td>Zelle 3/2</td>
  </tr>
  <tr>
    <td>Zelle 1/3</td>
    <td>Zelle 2/3</td>
    <td>Zelle 3/3</td>
  </tr>
</table>

</body>
</html>
```



# Tabelle ohne Gitternetzlinien

Zelle 1/1 Zelle 2/1 Zelle 3/1  
Zelle 1/2 Zelle 2/2 Zelle 3/2  
Zelle 1/3 Zelle 2/3 Zelle 3/3

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Tabelle ohne Gitternetzlinien</title>
</head>
<body>

<table border="0">
  <tr>
    <td>Zelle 1/1</td>
    <td>Zelle 2/1</td>
    <td>Zelle 3/1</td>
  </tr>
  <tr>
    <td>Zelle 1/2</td>
    <td>Zelle 2/2</td>
    <td>Zelle 3/2</td>
  </tr>
  <tr>
    <td>Zelle 1/3</td>
    <td>Zelle 2/3</td>
    <td>Zelle 3/3</td>
  </tr>
</table>

</body>
</html>
```

# Weitere Möglichkeiten

- HTML bietet noch viel mehr
  - Präformatierten Text (`<pre> ... </pre>`)
  - Zitate (`<blockquote> ... </blockquote>`)
  - Trennlinien (`<hr>`)
  - Verbinden von Zellen in Tabellen (`colspan`, `rowspan`)
  - Formulare (`<form action="..." method="..."> ... </form>`)
  - Frames, um Webseiten in mehrere Bildschirmfenster aufzuteilen
  - Einbindung dynamischer Inhalte mit JavaScript
  - Cascading Stylesheets (CSS), um HTML-Elemente exakt zu formatieren und positionieren
    - CSS ist eine einfache Möglichkeit um Inhalt und Darstellung zu trennen
  - ...

Sehr gute Quelle: <http://de.selfhtml.org>

# HTML und SGML

- Die HTML-Elemente sind der Metasprache SGML entliehen
  - SGML = Standard Generalized Markup Language
  - Mit SGML kann man verschiedene Auszeichnungssprachen definieren
    - Bekannte auf SGML basierenden Sprachentwicklungen: HTML und XML
- SGML **trennt Inhalt und Layout** eines Dokuments voneinander
  - Man kann somit beides unabhängig voneinander ändern
- Um ein gültiges SGML-Dokument zu beschreiben, verwendet man eine Dokumenttypdefinition (DTD), die den strukturellen Aufbau beschreibt
- Die Gültigkeit des Dokumentes überprüft man mit Parsern
  - Diese überprüfen, ob das Dokument konform mit der Deklaration im Header und der DTD ist
- HTML bis einschließlich HTML 4.01 ist eine Ableitung bzw. Anwendung von SGML
- XML ist eine Untermenge von SGML
  - In der Praxis hat XML heute den Platz von SGML eingenommen
- XHTML ist eine Neuformulierung von HTML 4.01 in XML

# XML – Erweiterbare Auszeichnungssprache

- Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdaten
- Wird u.a. für plattform- und implementationsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt
- Februar 1998: Erste Ausgabe der XML-Spezifikation
- XHTML und spätere Versionen genügen den Syntaxregeln von XML
- Die Namen der Strukturelemente (XML-Elemente) sind frei wählbar
- Ein Grundgedanke hinter XML ist die Trennung von Daten und ihrer Repräsentation
  - Ein XML-Dokument enthält keine Informationen darüber, wie der Inhalt dargestellt wird
  - Ziel: Eine Datenbasis im XML-Format für verschiedene Darstellungen
  - Beispiel: Daten sollen einmal als Tabelle und einmal als Grafik ausgegeben werden

Quelle: Wikipedia

# Wohlgeformtheit von XML-Dokumenten

- Ein XML-Dokument ist **wohlgeformt** (*well-formed*), wenn es alle XML-Regeln einhält
- Beispiele:
  - Das Dokument besitzt genau ein Wurzelement
    - Das Wurzelement ist das äußerste Element
    - Beispiel: `<html>` bei XHTML
  - Alle Elemente mit Inhalt besitzen einen Auszeichner (Tag) für Beginn und Ende
    - Beispiel: `<p> ... </p>`
    - Elemente ohne Inhalt können auch in sich geschlossen sein, z.B. `<br />`
  - Die Beginn- und End-Auszeichner sind ebenentreu-paarig verschachtelt
    - Es müssen alle Elemente vor dem End-Auszeichner des Elternelements geschlossen werden
  - Kein Element darf mehrere Attribute mit demselben Namen beinhalten

# Beispiel für eine einfache XML-Datei mit DTD

- Interne DTD
- Darstellung im Firefox Browser

Mit dieser XML-Datei sind anscheinend keine Style-Informationen verknüpft. Nachfolgend wird die Baum-Ansicht des Dokuments angezeigt.

```
-<webadressen>
-  <adresse>
    <url>http://www.h-da.de</url>
    <name>Hochschule Darmstadt</name>
  </adresse>
-  <adresse>
    <url>http://www.hs-mannheim.de</url>
    <name>Hochschule Mannheim</name>
  </adresse>
</webadressen>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE webadressen [
  <!ELEMENT webadressen (adresse)*>
  <!ELEMENT adresse (url, name)>
  <!ELEMENT url (#PCDATA)>
  <!ELEMENT name (#PCDATA)>
]>
<webadressen>
  <adresse>
    <url>http://www.h-da.de</url>
    <name>Hochschule Darmstadt</name>
  </adresse>
  <adresse>
    <url>http://www.hs-mannheim.de</url>
    <name>Hochschule Mannheim</name>
  </adresse>
</webadressen>
```

- Alternativ verwendet man eine externe Dokumenttypdefinition (DTD)

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE webadressen SYSTEM "webadressen.dtd">
```

# Beispiel für eine einfache XML-Datei (CD-Katalog)

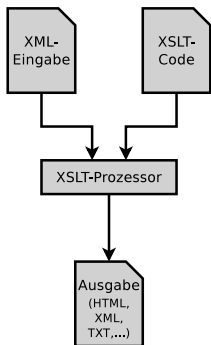
## ● Darstellung im Firefox Browser

Mit dieser XML-Datei sind anscheinend keine Style-Informationen verknüpft. Nachfolgend wird die Baum-Ansicht des Dokuments angezeigt.

```
-<CATALOG>
- <CD>
  <TITLE>Never Stop</TITLE>
  <ARTIST>The Bad Plus</ARTIST>
  <COUNTRY>USA</COUNTRY>
  <COMPANY>Universal</COMPANY>
  <PRICE>14,90</PRICE>
  <YEAR>2010</YEAR>
</CD>
- <CD>
  <TITLE>Volume Two</TITLE>
  <ARTIST>Sonny Rollins</ARTIST>
  <COUNTRY>USA</COUNTRY>
  <COMPANY>Blue Note</COMPANY>
  <PRICE>9,90</PRICE>
  <YEAR>1999</YEAR>
</CD>
- <CD>
  <TITLE>Dakar</TITLE>
  <ARTIST>John Coltrane</ARTIST>
  <COUNTRY>USA</COUNTRY>
  <COMPANY>Universal</COMPANY>
  <PRICE>10,90</PRICE>
  <YEAR>2008</YEAR>
</CD>
</CATALOG>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CATALOG>
  <CD>
    <TITLE>Never Stop</TITLE>
    <ARTIST>The Bad Plus</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Universal</COMPANY>
    <PRICE>14,90</PRICE>
    <YEAR>2010</YEAR>
  </CD>
  <CD>
    <TITLE>Volume Two</TITLE>
    <ARTIST>Sonny Rollins</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Blue Note</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>1999</YEAR>
  </CD>
  <CD>
    <TITLE>Dakar</TITLE>
    <ARTIST>John Coltrane</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Universal</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>2008</YEAR>
  </CD>
</CATALOG>
```

# Transformation einer XML-Datei mit XSLT



- XSLT dient zur Definition von Umwandlungsregeln
- XSLT-Programme (XSLT-Stylesheets) sind selbst nach den Regeln des XML-Standards aufgebaut
- XSLT-Stylesheets in XML-Dokumente einbinden:

```
<?xml-stylesheet type="text/xsl" href="simple.xsl"?>
```

- Stylesheets werden von XSLT-Prozessoren eingelesen
  - Diese wandeln nach den den Anweisungen ein oder mehrere XML-Dokumente in das gewünschte Ausgabeformat um
- XSLT-Prozessoren sind in vielen modernen Webbrowsers integriert
  - Firefox
  - Opera (ab Version 9)
  - Internet Explorer (ab Version 6)



## Beispiel für ein XSLT-Stylesheet

- Dieses XSLT-Stylesheet wird im Anschluss auf das Beispiel einer einfachen XML-Datei (CD-Katalog) angewendet

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<html xsl:version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns="http://www.w3.org/1999/xhtml">
  <body style="font-family:Arial;font-size:14pt;background-color:#EEEEEE">
    <xsl:for-each select="CATALOG/CD">
      <div style="background-color:teal;color:white;padding:5px">
        <span style="font-weight:bold"><xsl:value-of select="ARTIST"/></span>
        - <xsl:value-of select="TITLE"/>
      </div>
      <div style="margin-left:5px;margin-bottom:1em;font-size:12pt">
        <xsl:value-of select="COMPANY"/>
        <span style="font-style:italic">
          (<xsl:value-of select="YEAR"/>)
        </span>
      </div>
    </xsl:for-each>
  </body>
</html>
```

Quelle: [http://www.w3schools.com/xml/xml\\_examples.asp](http://www.w3schools.com/xml/xml_examples.asp)

# XSLT-Stylesheet auf XML-Datei (CD-Katalog) angewendet

- Darstellung im Firefox Browser

## The Bad Plus - Never Stop

Universal (2010)

## Sonny Rollins - Volume Two

Blue Note (1999)

## John Coltrane - Dakar

Universal (2008)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="simple.xsl"?>
<CATALOG>
  <CD>
    <TITLE>Never Stop</TITLE>
    <ARTIST>The Bad Plus</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Universal</COMPANY>
    <PRICE>14,90</PRICE>
    <YEAR>2010</YEAR>
  </CD>
  <CD>
    <TITLE>Volume Two</TITLE>
    <ARTIST>Sonny Rollins</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Blue Note</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>1999</YEAR>
  </CD>
  <CD>
    <TITLE>Dakar</TITLE>
    <ARTIST>John Coltrane</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Universal</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>2008</YEAR>
  </CD>
</CATALOG>
```

# Nächste Vorlesung

Nächste Vorlesung:  
**12.1.2012**