Prof. Dr. Th. Letschert

THM
TECHNISCHE HOCHSCHULE MITTELHESSEN

CS1024

Internetbasierte Systeme

Bachelor of Science (Informatik)

HTML: Hypertext Markup Language
CSS: Cascading Stylesheets

HTML - Übersicht

HTML = Hypertext Markup Language

- ***Hypertext Markup Language***

  Markup **= Auszeichnung**

  Textelemente mit struktureller statt inhaltlicher Bedeutung

  Hypertext **= Text mit „verlinkter“ Struktur**

  Text besteht aus einem Netz von Dokumenten

- **(Klar-) Textformat mit eingestreuten Auszeichnungen (Tags)**
- **HTML-Seite / HTML-Dokument: Datei mit Text im HTML-Format**
- **Grundprinzip: Trennung von Inhalt, Struktur und Darstellung**

  Inhalt und Struktur (als Tags) sind Bestandteil des Dokuments, werden vom Server geliefert

  Darstellung wird vom Browser generiert

  Nur unvollkommen umgesetzt / umsetzbar

- **Das „Web“: Menge an Anwendungen die via HTTP kommunizieren**

  das sind (immer noch) im Wesentlichen Web-Sites (Web-Auftritte): HTML-Seiten die von Servern an Browser ausgeliefert werden

Beispiel 1: Eine sehr einfache Seite

```
<!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=UTF-8">
<title>Simple HTML</title>
</head>
<body>
<H1>Ein einfaches Html Dokument</H1>
Ein einfaches Html-Dokument enthält Text mit ein paar Tags. Wie etwa:
<ul>
<li>&lt;ol> oder</li>
 <li>&lt;li> oder</li>
</ul>
<p>
Das Ganze ist sehr einfach strukturiert, im Gegensatz zu einem
professionellen Dokument wie beispielsweise die Seite von
<A HREF="http://www.spiegel.de/">Spiegel Online</A>.
</body>
</html>
```

HTML - Übersicht

Beispiel 2: http://www.w3.org/TR/html401/ (gekürzter Anfang)

HTML Spezifikation , Version 4.01
Format: HTML 4

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html lang="en">
<!-- $Id: cover.html,v 1.2 1999/12/24 23:37:45 ijacobs Exp $ -->
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
  <title>HTML 4.01 Specification</title>
  <!-- Changed by: Ian B. Jacobs,  6-Dec-1998 -->
  <link rel="next" href="about.html">
  <link rel="contents" href="cover.html#minitoc">
  <link rel="stylesheet" type="text/css" href= "http://www.w3.org/StyleSheets/TR/W3C-REC">
  <link rel="STYLESHEET" href="style/default.css" type="text/css">
</head>
<body>
  <div class="navbar" align="center"> <a href="about.html">next</a>  
    <a href="#minitoc">table of contents</a>   <a href="index/elements.html">elements</a> 
    <a href="index/attributes.html">attributes</a>   <a href="index/list.html">index</a>
    <hr>
  </div>
  <div class="head">
    <p><a href="http://www.w3.org/"><img height="48" width="72" alt="W3C"
                                    src= "http://www.w3.org/Icons/w3c_home"></a></p>
    <h1>HTML 4.01 Specification</h1>
    <h2>W3C Recommendation 24 December 1999</h2>
    <hr title="Separator from Header">
    <h2>Abstract</h2>
    <p>This specification defines the HyperText Markup Language (HTML),  ...
    </p>
    …
</body>
</html>
```

Beispiel 3: http://www.w3.org/ (gekürzter Anfang)

Homepage W3C
Format: XHTML 1.0

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <!-- Generated from data/head-home.php, ../../smarty/{head.tpl} -->
  <head>
    <title>World Wide Web Consortium (W3C)</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <link rel="Help" href="/Help/" />
    <link rel="stylesheet" href="/2008/site/css/minimum" type="text/css" media="handheld, all" />
    <style type="text/css" media="print, screen and (min-width: 481px)">
      /*<![CDATA[*/
      @import url("/2008/site/css/advanced");
      /*]]>*/
    </style>
  </head>
  <body id="www-w3-org" class="w3c_public w3c_home">
     <div id="w3c_container">
     <!-- Generated from data/mast-home.php, ../../smarty/{mast.tpl} -->
     <div id="w3c_mast"><!-- #w3c_mast / Page top header -->
     <h1 class="logo">
     <a tabindex="2" accesskey="1" href="/">
        <img src="/2008/site/images/logo-w3c-mobile-lg" width="90" height="53" alt="W3C" />
     </a>
     <span class="alt-logo">W3C</span></h1>
     ...
</body>
</html>
```

Beispiel 4: http://www.w3.org/html/wg/ (gekürzter Anfang)

Seite der HTML Arbeitsgruppe im W3C Consortium
Format: HTML 5

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <title>W3C HTML Working Group</title>
    …
</head>
<body>

<div id="wrapper">

<!-- header beg -->
<div id="header">
<h1><span class="logo"><a href="/"><img
      src="/Icons/WWW/w3c_home_nb" alt="W3C" height="48"
      width="72" /></a></span><span id="headertext"><a
      href="..">HTML</a> » HTML Working Group</span></h1>
</div>
<!-- header end -->
<!-- content beg -->
<div id="content">

    <!-- sidebar beg -->
    <aside id='sidebar'>
    …
    </aside>
    <!-- sidebar end -->
    …
</body>
</html>
```

W3C Consortium: Herstellerunabhängige Standardisierung von Web-Technologien.

HTML Standards: 3 relevante W3C Standards, und jeder Browser-Hersteller macht letztlich was er will.

HTML/HTTP Historie

1989 Tim Berners-Lee, CERN, Genf: HTTP und HTML

gelangweilter Physik-Forscher mit unbenutztem Next-Computer definiert HTML und HTTP implementiert HTTP-Server und Text-basierten Browser

1991 - 1993 Marc Andreesen, Mosaic

Junger, gelangweilter Informatiker am NCSA an der University of Illinois erkennt das Potential von HTML und HTTP und implementiert einen graphischen Browser, NCSA-Mosaic genannt.

1994 Marc Andreesen, Jim Clark: Netscape

Gründung der Firma Netscape, Portierung des Browsers auf Windows, Umbenennung des Browsers in Communicator, Rechtsstreitigkeiten mit NCSA

FH Giessen-Friedberg Webserver online (ca. 650 Servern weltweit)

W3C Gründung

MS erwirbt Mosaic-Lizenz vermarket modifiziertte Mosaic/Comunicator Versionen als Internet Explorer

1996 Microsoft eröffnet den 1. Browser-Krieg

Dez. 1995 MS-Aktie fällt (erstmals) um 7% (= 2.5 Milliarden $)
Februar 1996 MS stellt 2500 Entwickler ein, verbessert IE, integriert TCP/IP in BS
Aggressive und erfolgreiche IE Vermarktung gegen Netscape

2003 Der 2. Browser-Krieg

Schlechte Qualität und Monokultur machen IE zu einem hohen Sicherheitsrisiko

Alternative Browser verbreiten sich, HTML-Standards beginnen sich durchzusetzen

http://www.w3.org

W3C – Das *World Wide Web Consortium*

The W3C mission is to lead the World Wide Web to its full potential by developing protocols and guidelines that ensure the long-term growth of the Web.

1994 gegründet von **Tim Bernes-Lee**

Gremium zur Standardisierung von **Web-Technologien**

Standards : Empfehlungen ohne verpflichtenden Charakter, erarbeitet in einbem öffentlichen Prozess durch Arbeitsgruppen

Organisationsstruktur

- **Host-Organisationen**
  MIT (USA), ERCIM (Europa), Kioto Univ. (Japan)
- **Mitglieds-Organisationen**
  ca. 350 kommerzielle Firmen und staatl. Organisationen
- **Working Groups**
  temporäre Arbeits-Gruppen für ein spezielles Thema
- **Coordination Groups**
  permanente Arbeitsgruppen für eine Themengebiet (z.B. HTML)
- **Administration**
  Direktor, CEO, Advisory Committee (Mitgliederversammlung), Advisory Board (gewählte Vertreter, Management), Technical Architecture Group (gewählte Vertreter Technik)

HTML - Übersicht

HTML-Varianten wichtige HTML-Spezifikationen sind:

HTML 1992 – historisch –
Ur-Version, nicht standardisiert, rein text-basiert

HTML 2 1995 – historisch –
erste standardisierte Version

HTML 3.2 1998 – historisch –
Formulare, Tabellen, Stylesheets

HTML 4.01 1999 – aktuell –
weit verbreiteter aktueller Standard

XHTML 1.0 2000 – aktuell –
Redefinition von HTML 4.01 auf Basis von XML

XHTML 1.1 2001 – aktuell –
bereinigte Version von XHTML

HTML5 20*??* – aktuell –
Redefinition / Erweiterung von HTML 4.01 und XHTML, HTML der Zukunft (?)
Entwurf liegt vor, wird von einigen Browsern teilweise unterstützt

HTML und XHTML

aktuell verwendete HTML-Varianten sind:

HTML

in Form von **HTML 4.01 (1999)**

Spezifikation: http://www.w3.org/TR/html401/

XHTML

in Form von **XHTML 1.0 (2000, 2002)**

XHTML 1.1 gilt als nicht verwendbar da nicht IE-kompatibel

Spezifikation: http://www.w3.org/TR/xhtml1/

XHTML ist weniger beliebig als HTML

- Alle Tags müssen abgeschlossen sein
- Alle Element klein geschrieben
- Alle Attribute in Anführungszeichen
- …

XHTML basiert auf XML

- Kann darum von XML-Werkzeugen verarbeitet werden

Beobachtungen

Historische Webseiten findet man unter:
http://www.archive.org/index.php

Flexibilität von HTML und seiner Nutzer

Die weithin verwendete HTML-Versionen sind fast 10 Jahre alt und in den letzten 15 Jahren kaum verändert worden.

Die Web-Seiten von heute unterscheiden sich aber fundamental von denen vor 15 Jahren.

Kreatives Chaos

Im Unterschied zu „richtigen“ Programmiersprachen gibt es für HTML keine wirklich verbindliche Syntax. Browser schlucken stets möglichst viel, die Versions-Information wird dabei kaum beachtet.

Browser-Hersteller und W3C

Die Browser-Hersteller konkurrierten anfangs mit über eigene HTML-Elemente. Das W3C hat dem durch Standardisierung entgegen gewirkt.

Die Browser-Kriege mit Hilfe proprietärer HTML-Erweiterung haben glücklicherweise ein Ende gefunden, aber die treibende Kraft sind immer noch eher die Browser-Hersteller als das W3C.

Web-Programmierung zu einem großen Teil die Wissenschaft von den IE-Versionen.

Nächste HTML-Version

Momentan spricht einiges (nämlich Browser-Unterstützung) dafür dass HTML5 der nächste HTML-Standard sein wird.

HTML-Parser

Parser

analysiert Text, stellt Textstruktur fest, speichert interne (Baum-) Struktur

basiert auf einer Syntax-Definition

Syntax-Definition meist in Form einer Grammatik

HTML-Parser

Bestandteil des Browsers

Komplexe Software

Akzeptiert die sehr vielen fehlerhaften HTML-Dokumente (irgerndwie)

Tag Soup Parser : Akzeptiert i.d.R. mehrere (eventuell vermischte) Varianten

HTML und XHTML

HTML 4.01 Doctype in drei Varianten:

- **Strict** strikt dem Standard entsprechend

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

- **Transitional** wie strict, kann aber (im Jahr 2000) als veraltet erklärte HTML-Elemente enthalten

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

- **Frameset** wie transitional, kann aber Frames enthalten

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

Mime Type Server sollen Dokumente ausliefern mit dem Header

```
Content-Type: text/html;
```

SPIEGEL ONLINE

http://www.spiegel.de/

HTML und XHTML

Beispiel Spiegel online

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "
http://www.w3.org/TR/html4/loose.dtd">
<html>
  <!-- TE -->
  <head>
     <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
     <meta http-equiv="X-UA-Compatible" content="IE=8" />
      ...
  </head>
  <body> ...
  </body>
</html>
```

Dieses Dokument soll nach Art des IE8 (also standard-konform) dargestellt werden.

```
Date: Wed, 01 Sep 2010 09:33:09 GMT
Server: Apache-Coyote/1.1
X-Powered-By: Servlet 2.4; JBoss-4.0.3SP1 (build: CVSTag=JBoss_4_0_3_SP1
              date=200510231054)/Tomcat-5.5
Cache-Control: max-age=120
Expires: Wed, 01 Sep 2010 11:35:09 CEST
X-Host: lnxp-2863
Content-Type: text/html;charset=ISO-8859-1
Content-Length: 155620
X-Cache: MISS from www.spiegel.de, HIT from lnxp-1279.srv.mediaways.net
Age: 54
Via: 1.1 www.spiegel.de, 1.0 lnxp-1279.srv.mediaways.net:80 (squid/2.6.STABLE20)
Connection: keep-alive
```

Response Header

HTML und XHTML

- **XML-Deklaration:** XHTML Dokumente benötigen vorangestellte XML-Deklaration

```
<?xml version="1.0"?>
```

- **XHTML 1.0 Doctype** in drei Varianten entsprechend HTML 4.01
  - **Strict**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

  - **Transitional**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

  - **Frameset**

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

- **Namespaces** Das Wurzelelement muss eine Namespace-Deklaration enthalten

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

- **Mime Type** Server sollen Dokumente ausliefern mit dem Header

```
Content-Type: application/xhtml+xml  oder
Content-Type: text/html;
```

HTML und XHTML

http://www.w3.org/TR/xhtml1

Beispiel W3C XHTML 1.0 Spezifikation

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
  <meta name="generator" content="HTML Tidy, see www.w3.org" />
  <title>XHTML 1.0: The Extensible HyperText Markup Language (Second
         Edition)
  </title>
  ...
  </head>
  <body>
  ...
  <body>
</html>
```

```
Date: Wed, 01 Sep 2010 09:13:58 GMT
Server: Apache/2
Last-Modified: Thu, 01 Aug 2002 13:56:02 GMT
Etag: "1175a-3a726d575e080"
Accept-Ranges: bytes
Content-Length: 71514
Cache-Control: max-age=21600
Expires: Wed, 01 Sep 2010 15:13:58 GMT
P3p: policyref="http://www.w3.org/2001/05/P3P/p3p.xml"
Content-Type: text/html; charset=utf-8
```

! →

Response Header

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

```
Content-Type: text/html; charset=utf-8
```

Zeichencodierung

HTML Standards und Zeichenkodierung

Ein HTML-Dokument besteht aus Text in einer (fast) beliebigen Kodierung

Der Zeichencode sollte im HTML-Dokument angegeben werden

Der Webserver sollte dem Browser die verwendete Kodierung mitteilen

Immer in allen HML-Dokumenten den Zeichensatz korrekt definierten

Korrekt = so wie die Zeichen (z.B. vom Editor) abgespeichert werden

Nicht (!) etwa so wie sie gesendet werden sollten (Info, keine Anweisung)

Zeichen werden immer so gesendet wie der Server sie findet

Realität und Zeichencodierung

Die Angabe des Zeichensatzes im Dokument fehlt oft

Der Server sendet gelegentlich keinen entsprechenden Response-Header

Der Browser erwartet gelegentlich keine entsprechenden Informationen und wenn sie kommen dann ignoriert er sie auch mal.

Zeichencodierung

für Zeichen die im Zeichencode des Dokuments nicht darstellbar sind

Zeichencodierung mit Namen

Für eine Vielzahl von Zeichen gibt es eine HTML-Codierung in Form eines Namens

Beispiele: `&auml;` ~> ä `&euro;` ~> €

Numerische Zeichencodierung

Für alle Zeichen kann eine numerische Codierung verwendet werden

`&#`*<Nummer nach Unicode>*`;`

http://www.unicode.org/

Beispiel: `&#228;` ~> ä `&#8364;` ~> €

Namen und numerischer Code bestehen aus ASCII-Zeichen und sind damit wohl in jedem Zeichensatz vertreten.

Reservierte (HTML-) Zeichen

Einige Zeichen sind für HTML reserviert und müssen codiert werden, wenn sie im Text auftauchen

`<` ~> `<`
`>` ~> `>`
`&` ~> `&`

Das Zeichen, dass Sie beim Editieren eines HTML-Dokuments sehen, ist nur dann auch das Zeichen, das im Browser dargestellt wird, wenn der Editor es in dem Code abspeichert, mit der Browser es darstellt.

Struktur: Grundstruktur

Grundstruktur gilt für alle Varianten von HTML

DTD

Inhalt

Dokumenttyp Deklaration (DTD)

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

Dokument

```
<html>
```

Header

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

Body

```
<body>
  <H1>HTML</H1>
  <H2 align="center">Ist HTML einfach?</H2>
  <p>HTML ist nur auf den ersten Blick einfach, wie alles was aus
   einfachen Anf&auml;ngen in dynamischem Wildwuchs entstanden ist.
  </p>
  <H2 align="center">Selfhtml</H2>
  <p>Details können bei <a href="http://selfhtml.org">Selfhtml</a>
     nachgelesen werden!</p>
</body>
</html>
```

Struktur: Elemente und Tags

- **Hierarchische (Baum-) Struktur**

  Ein HTML-Dokument besteht aus (ineinander) verschachtelten Elementen

- **Tags (Auszeichnungen)**

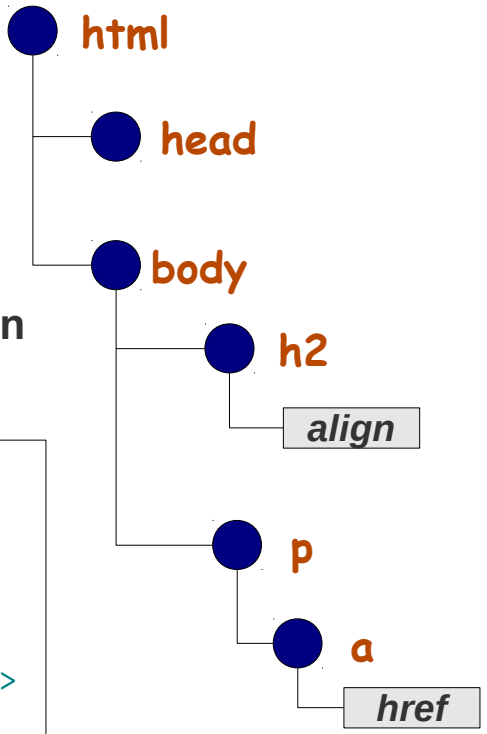
  begrenzen Elemente

  sind in spitze Klammern eingeschlossen

  haben **Namen** und eventuell

  Attribute

  Attribute sind Zuordnungen von Namen zu (String-) Werten




```
<html>
<head> <titel>HTML</titel></head>
<body>
<H2 align="center">Selfhtml</H2>
<p>Details können bei
  <a href="http://selfhtml.org">Selfhtml</a>
  nachgelesen werden!
</p>
</body>
</html>
```

HTML – Syntax : Header

http://de.selfhtml.org/html/kopfdaten/meta.htm

HTML Kopf `<head> ... </head>`

Unterelement (Bestandteile) des Headers sind:

- **Titel** (zwingend) `<title> ... </titel>`

  Titel des Dokuments (Name des Browser-Fensters, Name des Lesezeichens)

- **Metadaten** (optional) `<meta> ... </meta>`

  diverse Metainformation (Info über das Dokument) für den Server, Browser, Suchmaschinen, etc.

  Beispiel:

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<meta http-equiv="X-UA-Compatible" content="IE=8" />
<meta name="author" content="SPIEGEL ONLINE, Hamburg, Germany" />
<meta name="copyright" content="SPIEGEL ONLINE, Hamburg, Germany" />
<meta name="email" content="spiegel_online@spiegel.de" />
<meta name="robots" content="index, follow, noarchive" />
<meta name="MSSmartTagsPreventParsing" content="true" />
<meta http-equiv="imagetoolbar" content="no" />
<meta name="keywords" content="SPIEGEL ONLINE, DER SPIEGEL, Nachrichten, News,Home" />
<meta name="description" content="Deutschlands führende Nachrichtenseite. ..." />
<meta name="verify-v1" content="2lPZiw22dOKqJU1fIWXbiUzm3T/qiHzJlBL+87ZOfF8=" />
<meta name="msvalidate.01" content="0EE91CCF2745FAE5C53BFE9A010D3C79" />
```

HTML – Syntax : Elemente

Blockelemente

strukturieren das Dokument

erzeugen in der Darstellung einen Absatz (beanspruchen die gesamte Fensterbreite)

dürfen i.A. Text und weitere Elemente enthalten, d.h. sie dürfen i.d.R. verschachtelt werden

Manche dürfen nicht verschachtelt werden (z.B. h1 … h6)

- Beispiele

  `address, blockquote, del, div, dl, form, h1 .. h6, hr, ins, li, ol, p, pre, table, ul`

Inline-Elemente

sind immer Bestandteil von Text oder anderen Elementen

erzeugen in der Darstellung keinen Absatz (Einbettung in aktuelle Zeile)

dürfen i.A. nur andere Inline-Elemente enthalten

werden i.d.R. nicht verschachtelt

- Beispiele

  `a, abbr, acronym, b, bdo, big, br, button, cite, code, del, dfn, i, img, ins, input, kbd, label, map, object, q, samp, script, select, small, span, strong, sub, sup, textarea, tt, var`

HTML – Syntax : Blockelemente

Blockelemente

wichtige Blockelemente sind

`h1, h2, ...`	Überschriften
`br`	Zeilenumbruch
`p`	Paragraph
`ul,ol,li`	Listen und Listenelemente
`table`	Tabellen
`hr`	horizontale Linie
`blockcode`	Zitat
`div`	Abschnitt/Bereich (vor allem für CSS-Strukturierungen)

HTML – Syntax : Blockelemente

Inline-Elemente

wichtige Inline-Elemente sind

strong	stark betont
em	betont
tt	Teletype (nicht-proportionale Schrift)
sub	tiefer gestellt
cite	Zitat (in der Zeile)
span	Abschnitt/Bereich, in der Zeile (vor allem für CSS-Strukturierungen)

Attribute

Tags können mit Attributen versehen werden

Manche Attribute sind nur für bestimmte Elemente zugelassen

Universelle Attribute können für alle Elemente verwendet werden

Beispiel:

```
<!-- Navigation start -->
    <div id="navi" title:"Navigation">
      <h2 align="left">Entdecke mich</h2>
      <ul style="list-style-type:none">
         <li><a href ="/">Startseite</a></li>
         <li><a href ="/Verdienste">Meine Verdienste</a></li>
         <li><a href ="/Ehrungen">Meine Ehrungen</a></li>
      </ul>
    </div>
<!-- Navigation end -->
```

universelles Attribut

spezielles Attribut

universelles Attribut

Anker-Elemente

a Anker-Element mit Verweis (Link) auf anderes Dokument als Verweis und Text als Inhalt

href Attribut enthält den Hyper-Link in Form einer **URL**

Beispiel

```
<h3>
  <a href="/auto/fahrberichte/0,1518,722399,00.html"
       title="Bugatti Veyron 16.4 Super Sport: Das Über-Auto">
           Das Über-Auto
   </a>
</h3>
```

aus SpiegelOnline 12.10.210

Hyperlink / URL

`http` => Get-Request
Pfad beginnt mit *document-root,* dem Basis-Verzeichnis

`file` => Datei öffnen und anzeigen,
Pfad beginnt mit *document-root,* dem Basis-Verzeichnis

kein Schema relativer Pfad; Pfad der nicht mit **/** beginnt
ist Pfad relativ vom Verzeichnis des aktuellen Dokuments

`mailto` => Mail-Programm öffnen

`ftp` => FTP-Request

Beispiele

```
<a href="chapter2.html">chapter two</a>

<a href="../images/forest.gif">map of the enchanted forest.</a>

<a href="http://www.w3.org/">W3C Web site</a>.
```

HTML – Syntax : Anker und Links

Hyperlink / Anker und lokale Verweise

`<a id="name">` **Anker definieren**

`<a href="#name">` **Bezug auf den Anker**

Beispiel

```
<a name="chapter1">Kapitel 1</a>

<p>Kapitel 1 ist sehr spannend!</p>

<em>...jede Menge mehr HTML...</em>

In <a href="#chapter1">Kapitel 1</a> kann man den Anfang lesen!
```

Listen

`ul`	**nicht-nummerierte Liste**
`ol`	**nummerierte Liste *(ordered list)***
`li`	**Listenelement**
`dl`	**Definitionsliste**
`dt`	**Definition: Begriff *(definition term)***
`dd`	**Definitionsliste: Erklärung *(definition data)***

Beispiele

```
<dl>
  <dt>Coffee</dt> <dd>Black hot drink</dd>
  <dt>Milk</dt>   <dd>White cold drink</dd>
</dl>
```

Definitionsliste

```
<h4>Circle bullets list:</h4>
<ul type="circle">
 <li>Apples</li>
 <li>Bananas</li>
 <li>Lemons</li>
 <li>Oranges</li>
</ul>
```

nicht-numerierte Liste mit rundem „Blobbs“

Tabellen

`table`	Tablelle
`tr`	Zeile der Tabelle *(table row)*
`td`	Zelle einer Zeile *(table data)*
`th`	Spaltenüberschrift

Beispiele

```
<h4>Two rows and three columns</h4>
<table border="1">
<tr><td>100</td> <td>200</td> <td>300</td></tr>
<tr><td>400</td> <td>500</td> <td>600</td></tr>
</table>
```

2x3 Tabelle mit umrandeten Zellen

100	200	300
400	500	600

```
<h4>Table with headers:</h4>
<table border="0">
<tr>
  <th>Name</th><th>Telephone</th><th>Fax</th>
</tr>
<tr>
  <td>Bill Gates</td><td>555 77 854</td><td>555 77 855</td></tr>
  <tr><td>Bill Clinton</td><td>666 88 341</td><td>666 88 331</td>
</tr>
</table>
```

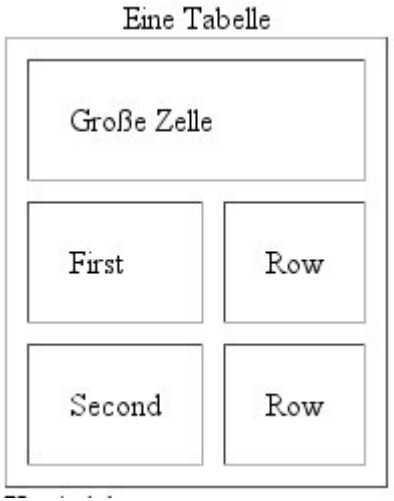
3x2 Tabelle mit Spaltenüberschriften und ohne Rand

Name	Telephone	Fax
Bill Gates	555 77 854	555 77 855
Bill Clinton	666 88 341	666 88 331

Tabellen

Beispiel

```
<table border="1" cellspacing="10" cellpadding="20">
  <caption>Eine Tabelle</caption>
  <tr>
      <td colspan="2">Große Zelle</td>
   </tr>
   <tr>
      <td>First</td>
      <td>Row</td>
   </tr>
   <tr>
     <td>Second</td>
     <td>Row</td>
   </tr>
</table>
```



HTML lernen

zu HTML existiert eine Vielzahl von Informationen und Lehrwerken, viele auch online.

Wir empfehlen

```
<a href="http://selfhtml.org">Selfhtml</a>

<a href="http://aktuell.de.selfhtml.org/weblog/selfhtml-wiki-online">Selfhtml Wiki</a>

<a href="http://www.w3schools.com/html/default.asp">W3C HTML Tutorial</a>
```

An HTML ist nur wenig lehrbar. Lernen Sie den Umgang mit HTML selbständig und selbst-bestimmt unter Verwendung Ihnen geeignet erscheinender Unterlagen !

Warum CSS: HTML Prinzip und Realität

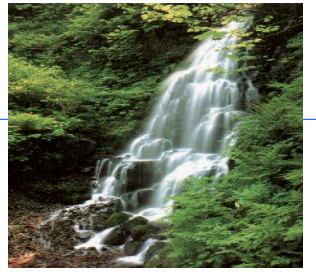
Prinzip: Trennung von Inhalt und Struktur-Informationen

```
<h1>Kapitel 1</h1>
Eine <em>wichtige</em> Sache!
<h2>Absatz 1.1</h2>
Ein Unterpunkt der wichtigen Sache.
```

Strukturinfo Text: Haupt- und Unterabschnitt.
Strukturinfo Zeichen: Wichtig.
Darstellung bleibt dem Browser überlassen.

Realität: HTML-Autoren wollen Darstellung beeinflussen

- HTML sollte die Struktur beschreiben, aus der der Browser dann das Layout ableitet
- Gute Idee – für reine Texte – also bis ca. 1992
- Schlechte Idee für stark strukturierte Seiten, der Autor will und muss Einfluss auf das Layout nehmen
- Erste „Lösung“:
  Erfindung von Layout-Tags, Missbrauch von Struktur-Tags, Einfügen von leeren Bildern, ...
- Lösung ab HTML 3: **Style Sheets** (Stilvorlagen)



CSS Prinzip

CSS: Cascading Style Sheets – Stufenförmige Stilvorlagen

Stilvorlagen beschreiben das Layout eines oder vieler Dokumente, also Ding wie:

- Textlayout: Schriftart und -Größe, Textfarbe, …
- Absatzbreite, Zeilenabstand
- Hintergrundfarben, Rahmenbreiten
- Layout von Tabellen und Formularen, …

Cascading Style Sheets

- Die Stilvorlagen werden stufenförmig, in einer definierten Reihenfolge, auf das Dokument angewendet

Wozu Style Sheets

- Praktikable Trennung von Layout und Inhalt
- Zentrale Layout-Steuerung (Ein Sheet für viele Dokumente)
- Flexibilität im Layout

CSS Beispiele

```
<body style="background-color: EEEEEE; font-family: sans-serif">
```

*ein **Style-Attribut**: helles Grau als Hintergrundfarbe und serifenlose Schrift im Body*

```
<html>
<head>
   <titel>CSS Example</titel>
   <style>
    <!--
        H1 { margin-left: 0.21cm; margin-right: 0.21cm;  font-family: Arial }
        A:link { color: #00008b; text-decoration: none;  font-family: Arial}
    -->
   </style>
</head> <body> ... </body>
</html>
```

*ein **Style-Tag**: Darstellung von H1-Elementen und Links*

Drei Stylesheets

Drei unterschiedliche Style Sheets können die Darstellung eines Dokuments beeinflussen:

Browser-Stylesheet Layout-Vorgabe des Browsers

- Jeder Browser hat sein eigenes Stylesheet, das er verwendet wenn es keine andere Vorgabe gibt

Autoren-Stylesheet Layout-Vorgabe des Autors des Dokuments

- Der übliche Fall: der Autor der Dokumente liefert ein Stylesheet

Benutzer-Stylesheet Layout-Vorgabe des Benutzers

- Der Benutzer kann mit seinem Stylesheet alle anderen überschreiben

Drei Verwendungsorte von Autoren-Stylesheets

Drei unterschiedliche Verwendungsorte für Style Sheets:

Style-Attribut

- Jedes HTML-Tag kann mit dem Attribut style versehen werden

Style-Tag

- Im Kopf des Dokumentes wird ein Stylesheet mit allen für das Dokument gültigen Definitionen platziert

Style-Datei (externes Stylesheet)

- Die Definitionen werden in einer Datei abgelegt so können verschiedene Dokumente sich darauf beziehen

Drei Verwendungsorte von Autoren-Stylesheets: Attribut und Tag

```
<body style="background-color: EEEEEE; font-family: sans-serif">
```

Style-Attribut

```
<html>
<head>
   <titel>CSS Example</titel>
   <style>
    <!--
         body { background-color: EEEEEE; font-family: sans-serif }
    -->
   </style>
</head> <body> ... </body>
</html>
```

Style-Tag

Drei Verwendungsorte von Autoren-Stylesheets: Externes Sheet

```
<html>
<head>
   <titel>CSS Example</titel>
   <link rel="stylesheet" type="text/css" href="style/myStyle.css">
</head> <body> ... </body>
</html>
```

Bezug auf externes Stylesheet

```
body {
    background-color: EEEEEE;
    font-family: sans-serif
}
```

Externes Stylesheet

Regeln

siehe Spezifikation:
http://www.w3.org/TR/CSS2/

Stylesheets enthalten Darstellungs-Regeln

Struktur der Regeln

***<Regel>* ::= *<Selektor>* { *<Deklaration>*; *<Deklaration>*; … }**

<Deklaration>* ::= *<Eigenschaft>*: *<Wert>

Selektor

wählt die Elemente aus auf die die Deklaration(en) angewendet werden soll. (Mustervergleich von Selektor und Element)

Deklaration

gibt Gestaltungsanweisungen durch Angabe einer Eigenschaft und deren Gestaltung

Selektoren / Übersicht

Arten von Selektoren :

Universalselektor	`*`	passt zu allen Elementen
Typselektor	`h1`	passt zu Elementen von diesem Typ
Id-Selektor	`#navi`	passt zu Elementen mit dieser Id
Klassenselektor	`.wichtig`	passt zu Elementen mit dieser Klasse

Selektor-Kombinationen

Gruppierung	`h1, h1`	ersetzt mehrere Regeln
Nachfahr-Selektor	`#navi ul`	passt zu Nachkommen- in Elternelement
Kind-Selektor	`p > i`	passt zu Kind- in Elternelement
Geschwister-Sel.	`p + p`	passt zu Geschwisterelementen
Attribut-Selektor	`img[alt]`	passt zu Elementen mit bestimmten Attribut

Pseudo-Klassen

Link im Orgi.-Zustand	`a:link`
Besuchter Link	`a:visited`
Maus auf Link	`a:hover`

Ungültige und syntaktisch falsche Selektoren werden ignoriert.

Regel-Beispiele

- `* { color: #2e2e2e }`
  Formatiert alle Texte dunkelgrau
- `div p { color: olive }`
  Formatiert alle Texte in einem <p>-Element innerhalb eines -Elements oliv
- `div * p { color: olive }`
  Formatiert alle Texte in einem <p>-Element innerhalb eines -Elements mit irgendeinem Element zwischen und <p> oliv
- `h1 { font-family: Arial, sans-serif; font-weight: bold }`
  Formatiert alle <h1>-Elemente
- `div#wrapper { position: relative; top: 10px; left: 10px }`
  Positioniert -Elemente mit der Id „wrapper“
- `h1, h2 div#footer { color: maron }`
  Formatiert alle <h1>- und <h2>-Elemente sowie alle -Elemente mit der Id „footer“

Regel-Beispiele

- `p em { color: red }`
  Formatiert alle -Elemente innerhalb eines <p>-Elements

- `p > em { color: red }`
  Formatiert alle -Elemente direkt innerhalb eines <p>-Elements

- `h2 + p { font-weight: bold }`
  Formatiert alle <p>-Elemente die direkt nach einem <h2>-Element auftreten

- ```
  img[alt] {
      padding: 2px;
      border:  1px;
      solid:   #000;
      background-color: #2e2e2e
  }
  ```
  Formatiert alle <img>-Elemente mit dem Attribut „alt“ (schwarzer Rahmen mit 2 Pixel Abstand und grauer Hintergrund zwischen Bild und Rahmen)

- `a:link { text-decoration:none }`
  Formatiert unbesuchte Links (keine Unterstreichung)
  ```

Kaskaden

Die Regeln „ergießen“ sich in einer bestimmten Reihenfolge über das Dokument (Stufen in in ansteigender Priorität):

1. **Browser**-Stylesheet
2. **Benutzer**-Stylesheet
3. **Autoren**-Stylesheet
4. **Autoren**-Stylesheet / Regeln mit der Markierung `!important`
5. **Benutzer**-Stylesheet / Regeln mit der Markierung `!important`

Regeln einer Stufe werden sortiert:

- `!important` überschreibt andere ebenfalls passende Regeln
- **Spezifischere** Regeln überschreiben weniger spezifischere
  z.B. style-Attribute eines Element überschreiben andere Regeln, Id-Attribute überschreiben Klassen-Attribute, die zuletzt gelesene Regel überschreibt eine vorherige, etc. (komplexe Regelung siehe Spezifikation)

Vererbung

Html-Elemente stehen in einer **Vererbungs**-Relation. Elemente denen über die Regel-Kaskade keinen Stil zugewiesen wurde, erhalten ein Stil von ihrem Vorgänger in der Vererbungshierarchie.

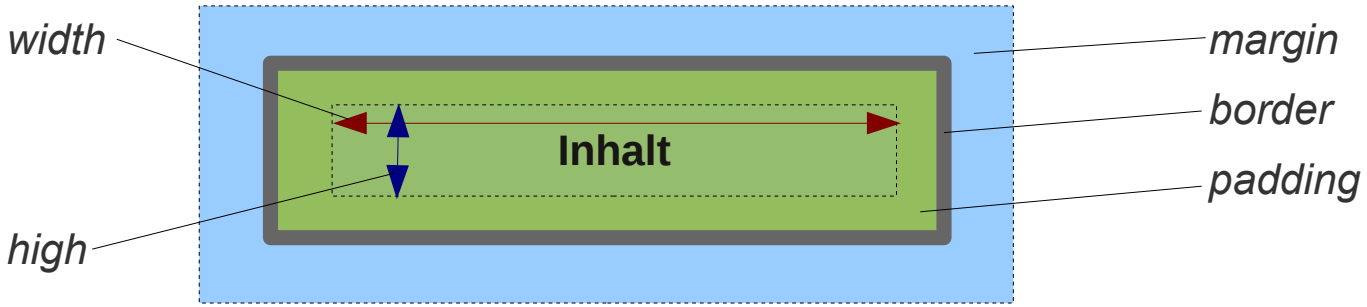
Die Vererbungs-Relation ergibt sich aus der **Dokumenten-Struktur**.

Nur **sinnvolle** Eigenschaften werden vererbt.

Nicht vererbt werden z.B.: Hintergrund-Eigenschaften (Bild, Farbe) und Positionierungen

Das Box-Modell

- **Box**
  Allen Elementen eines Dokuments wird eine Browser-Fläche („Box“) zugewiesen
- **Standard-Position einer Box**
  Boxen von Block-Elementen (z.B. `p, table, h1, hr`) werden untereinander angeordnet
  Boxen von Inline-Elementen (z.B. `a, br, cite, img, em`) werden nebeneinander angeordnet
- **Box-Eigenschaften**
  - Inhalt — Inhalt des Elements, z.B. Text, Bild
  - width — Breite des Inhalts
  - height — Höhe des Inhalts
  - padding — Innenabstand: Abstand zwischen Inhalt und Rahmen
  - border — Rahmen
  - margin — Außenabstand: Abstand zwischen Rahmen und anderen Boxen

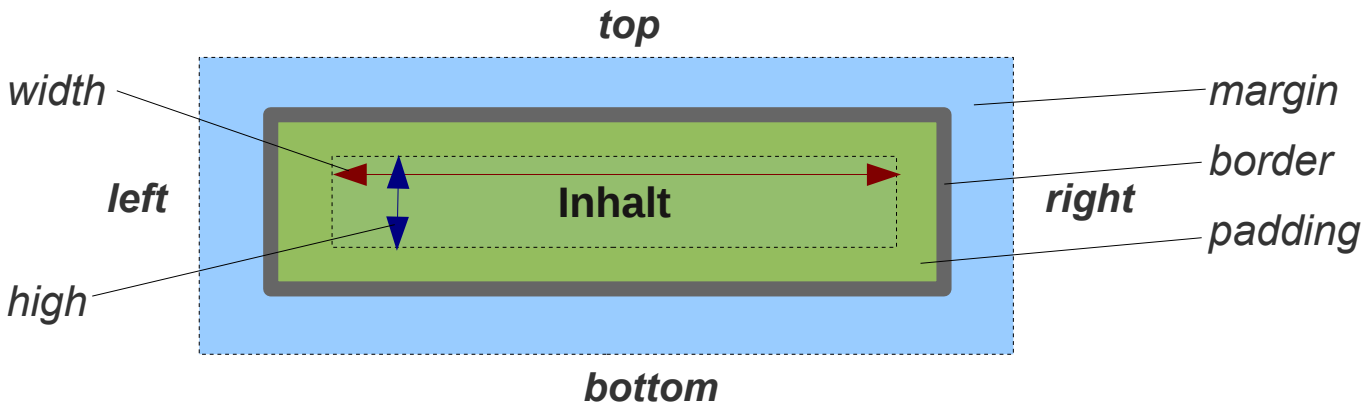


Das Box-Modell

- **Box-Eigenschaften definieren**

  Beispiel:

```
p {
  padding: 10px 20px 30px 40px;
  // = padding-top:10px; padding-right: 20px; padding-bottom: 30px; padding-left: 40px;
  margin:         30px;
  // = margin: 30px 30px 30px 30px;
  border:         5px solid silver;
  // = border-width: 5px; border-style: solid; border-color: silver;
}
```



Positionierungen

- **Klassische Form der Positionierung**
  Anordnung aller Elemente in einer Tabelle
  Kompliziert, vermischt Inhalt und Layout
- **Moderne Form der Positionierung**
  CSS-Positionierung
  Kompliziert, trennt Inhalt und Layout
- **CSS-Positionierung**
  Vorgehen
  - Html-Text
    ohne Positonions-Information
    mit Einteilung in inhaltliche Bereiche
    z.B.: `header, navi, content, footer`
  - Stylesheet
    positioniert die inhaltlichen Bereiche

Positionierungen – Vorgehen bei zweispaltigem Layout

- Erstelle HTML-Dokument mit der Grundstruktur des Textes
  zB. Kopf / Navigation / Inhalt / Fuß

```
<div id="header">...</div>
<div id="navi">...</div>
<div id="content">...</div>
<div id="footer">...</div>
```

  und fülle ihn mit Text

- Erstelle Stylesheet das jedem Bereich eine eigene Hintergrundfarbe gibt (macht die Positionierung deutlich)
- Positioniere die Navigation links
- Gib dem Inhalt einen großen linken *Margin*
- Definiere einen *Wrapper*-Div um Inhalt und Navigation um sie auf gleiche Höhe auszurichten
- Füge `clear:both` zu den Definitionen für den Fußbereich hinzu, damit wird ein Umfließen verhindert
- Setze die Farben der Bereiche derart, dass ein einheitliches Erscheinungsbild entsteht

Positionierung / Beispiel – HTML – 1

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
  <head>
    <meta name="language" content="german, de, deutsch"/>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <meta name="robots" content="index,follow" />
    <meta name="keywords" content="Genie, Ich">
    <meta name="DC.Publisher" content="Ich">
    <meta name="DC.rights" content="Alle Rechte liegen bei mir">
    <link rel="stylesheet" type="text/css" href="style/myStyle.css">
    <title>Ich</title>
  </head>
  <body>
```

```
<!-- Branding start -->
<div id="branding">
  <h1>Ich</h1>
  <blockquote>
  <p>Schönheit liegt im Auge des Betrachters und nur Selbstbetrachtung garantiert einen kompetenten Betrachter.</p>
  </blockquote>
</div>
<!-- Branding end -->
```

```
<!-- Wrapper start -->
<div id="wrapper">
```

```
<!-- Navigation start -->
<div id="navi">
  <h2>Entdecke mich</h2>
  <ul>
    <li><a href ="/">Startseite</a></li>
    <li><a href ="/Verdienste">Meine Verdienste</a></li>
    <li><a href ="/Ehrungen">Meine Ehrungen</a></li>
  </ul>
</div>
<!-- Navigation end -->
```

Positionierung / Beispiel – HTML – 2

```
<!-- Content start -->
<div id="content">
        <a href="/Verdienste/"><img src="images/ich.jpg" width="150" height="200"
title="Ich"
alt="Bild von mir"></a>
  <h3>Google, wo guckst Du? Such Mich!</h3>
  <ul>
    <li><h4>Design ist alles!</h4>
        <p>Lorem ipsum consectetuer adipiscing elit. </p>
        <p>Eget habitasse elementum est.</p>
    </li>
    <li><h4>Inhalt auch!</h4>
        <p>Hat man keinen dann braucht man Füllsel. Statt zu tippen suchen Sie im Internet nach
           <em>Blindtext</em>.</p>
    </li>
  </ul>
</div>
<!-- Content end -->
</div>
<!-- wrapper end -->
```

```
<!-- site info start -->
<div id="site_info">
  <h4>Siteinfo</h4>
  <address class="vcard">
      <span class="org">Ich Desgin, Inc.&copy;</span><br/>
      <span class="email">mein.ich@myself.com</span><br/>
      Alle Rechte vorbehalten, All Rights Reserved
  </address>
</div>
<!-- site info end -->
```

```
  </body>
</html>
```

Positionierung / Beispiel – CSS

```
body {
  background-color: teal;
}

blockquote {
  font-style: oblique;
  font-variant: small-caps;
  font-weight: bold;
  text-align:right;
  font-size:20px;
  color: #101010;
}

a:link { color: #001050; text-decoration: none;  font-family: Arial}
a:visited { color: #000080; text-decoration: none;  font-family: Arial}

#wrapper {
  width: 98%;
  background-color: teal;
}

#branding {
  font: Arial;
  color: #101010;
  padding-right: 10px;
  background-color: teal;
}

#site_info {
  clear: both;
  padding-right: 30px;
  background-color: teal;
}
```

Positionierung / Beispiel – CSS

```
#navi {
  margin: 0px;
  float: left;
  width: 130px;
  padding: 20px 40px 140px 20px;
  background-color: teal;
  font: Arial;
}

#navi li {
  font: Arial;
  font-variant: small-caps;
}

#content {
  padding-right: 10px;
  padding-left: 10px;
  padding-top: 10px;
  margin-left: 200px;
  background-color: silver;
}

#content p {
  color: #303030;
  font: Arial;
}

#content h3 {
  color:  #202020;
}

#content h4 {
  color: #202020;
}
```

Positionierung / Beispiel – Ergebnis

Ich

Schönheit liegt im Auge des Betrachters und nur Selbstbetrachtung garantiert einen kompetenten Betrachter.

Entdecke mich

- Startseite
- Meine Verdienste
- Meine Ehrungen

Google, wo guckst Du? Such Mich!

- **Design ist alles!**

  Lorem ipsum consectetuer adipiscing elit.

  Eget habitasse elementum est.

- **Inhalt auch!**

  Hat man keinen dann braucht man Füllsel. Statt zu tippen suchen Sie im Internet nach *Blindtext*.

Siteinfo

Weitere Informationen

http://www.w3schools.com/css/

Die Beherrschung von CSS erfordert Geduld und Leidensfähigkeit bzw. Begeisterung. Beides ist nicht lehrbar. Lernen Sie den Umgang mit CSS selbständig und selbst-bestimmt unter Verwendung Ihnen geeignet erscheinender Unterlagen!