

Digitale Medien

4. STYLESHEETS, CSS

Vergangene Vorlesungen

- HTML-Grundlagen
- Wichtige HTML-Elemente

- Heute: Stylesheets
- Anwendung
- Beispiele

Trennung von Inhalt und Layout

Oder: Woher weiß der Browser wie eine Seite darzustellen ist?

- HTML strukturiert nur
- trotzdem werden HTML-Elemente formatiert
- → Browser hat Default-Layout: Browser-Stylesheet

- Heute: Wie kann ich mein eigenes Design integrieren?

Style-/Formatangaben

- Stylesheets erlauben die Beschreibung der Formateigenschaften einzelner HTML Elemente
- Definition neuer Formate, die HTML Elementen zugewiesen werden
- drei Möglichkeiten:
 - separat für jedes HTML-Element
 - Style-Angaben werden als Attribut des jeweiligen HTML-Elements angegeben
 - Style-Angaben werden für eine bestimmte ID angegeben
 - durch <style></style> Element innerhalb des HTML-Dokuments
 - innerhalb einer Stylesheet Datei
- Alle drei Möglichkeiten können gleichzeitig verwendet werden
- -> Stylesheets können hintereinander geschaltet, kaskadiert werden
- -> Cascading Stylesheets (CSS)

Style Angabe für jedes HTML-Element


einfachster Fall

```
<html>
<head>
  <title>Titel der Datei</title>
</head>
<body>
  <h1>Überschrift</h1>
</body>
</html>
```

Style Angabe für jedes HTML-Element

einfachster Fall

```
<html>
<head>
  <title>Titel der Datei</title>
</head>
<body>
  <h1 style="color:red; font-size:48px;">
    48 Pixel und Rot!
  </h1>
</body>
</html>
```

- Angabe für jedes Element einzeln zugeordnet 

Stylesheets

7

- Besser: Formatangaben zentral in Style-Element

```
<html>
<head>
  <title>Titel der Datei</title>
  <style type="text/css">
    /*Formatangaben für HTML-Elemente*/
  </style>
</head>
<body>
  <h1>Überschrift</h1>
</body>
</html>
```

Stylesheets - Syntax

8

- Syntax eines Eintrags im Stylesheet:

```
Selektor {
  Eigenschaft:Wert;
}
```

- Selektor: HTML-Element (Tag-Name), bspw.: H1
- Eigenschaft: bspw. color, width, height, ...
- Wert: wird der Eigenschaft zugeordnet, bspw.: red, 200px,
- Selektor kann auch Wildcard * sein

Stylesheets im Style-Element

9

```
<html>
<head>
  <title>Titel der Datei</title>
  <style type="text/css">
    h1 { color:red; font-size:48px; }
  </style>
</head>
<body>
  <h1>48 Pixel und Rot!</h1>
</body>
</html>
```

Stylesheets Individualformate

10

- Für jedes HTML-Element kann eine eindeutige ID vergeben werden
 - Darf nur einmal im gesamten HTML-Dokument vorkommen

```
<html>
<head>
  ...
</head>
<body>
  <h1 id="Titel1">48 Pixel und Rot!</h1>
</body>
</html>
```

Stylesheets Individualformate

11

- IDs können für die Zuweisung von Formatangaben verwendet werden

```
<html>
<head>
  <title>Titel der Datei</title>
  <style type="text/css">
    #title1ID { color:red; font-size:48px; }
  </style>
</head>
<body>
  <h1 id="title1ID">48 Pixel und Rot!</h1>
</body>
</html>
```

Stylesheets Individualformate

12

- Verwendung des class-Attributs

```
<html>
<head>
  ...
</head>
<body>
  <h1 class="title">Ein bisschen Text</h1>
</body>
</html>
```

Formatangaben für Klassen

```
<html>
<head>
<title>Titel der Datei</title>
<style type="text/css">
  .title {font-size:18pt;color:Red;}
</style>
</head>
<body>
<h1 class="title">Überschrift 1</h1>
<p>Ein bisschen Fließtext</p>
<h1>Überschrift 2</h1>
<p>Ein bisschen Fließtext</p>
<h1 class="title">Überschrift 3</h1>
<p>Ein bisschen Fließtext</p>
</body>
</html>
```

Stylesheets

```

formate.css
14 .title {
    font-size:18pt;
    color:red;
}

seite.html
<html>
<head>
<title>Titel der Datei</title>
<link rel="stylesheet" type="text/css"
href="formate.css">
<style type="text/css">
  body { background-color:#FFFFCC;margin-left:100px;}
  ...
</style>
</head>
<body>
<div class="title">Dies ist der Titel</div>
</body>
</html>
```

Pseudoformate

- Formate für HTML-Bestandteile, die sich jedoch nicht durch ein eindeutiges HTML-Element ausdrücken lassen
 - Ein noch nicht besuchter Link
 - Ein besuchter Link
 - Der erste Buchstabe eines Absatzes

```
<style type="text/css">
a:link { font-weight:bold; color:#0000E0;}
a:visited { font-weight:bold; color:#000080;}
a:hover { font-weight:bold; color:#E00000;}
a:active { font-weight:bold; color:#E00000;}
a:focus { font-weight:bold; color:#00E000;}
p:first-line { font-weight:bold }
p:first-letter { font-size:300%; color:red;}
</style>
```

Beispiele für CSS-Eigenschaften

Schriftarten und -parameter

- ↓ Allgemeines zur Schriftformatierung
- ↓ font-family (Schriftart)
- ↓ font-style (Schriftstil)
- ↓ font-variant (Schriftvariante)
- ↓ font-size (Schriftgröße)
- ↓ font-weight (Schriftgewicht)
- ↓ font-stretch (Schriftlaufweite)
- ↓ font (Schrift allgemein)
- ↓ word-spacing (Wortabstand)
- ↓ letter-spacing (Zeichenabstand)
- ↓ text-decoration (Textdekoration)
- ↓ text-transform (Texttransformation)
- ↓ color (Textfarbe)
- ↓ text-shadow (Textschatten)

Beispiele für CSS-Eigenschaften

Hintergrundfarbe & -bilder

- Hintergrundfarbe: background-color

```
<html>
<head>
<title>Titel der Datei</title>
<style type="text/css">
  body { background-color:#FFFFCC;margin-left:100px;}
  h1 { font-size:300%;color:#00FF00;
font-style:italic;
border-bottom:solid thin black; }
  .text {font-size:18pt;color:Red;}
</style>
</head>
<body>
<h1>Überschrift</h1>
<div class="text">Dies ist der Text...</div>
</body>
</html>
```

Beispiele für CSS-Eigenschaften

Hintergrundfarbe & -bilder

- Hintergrundbild: background-image

```
<html>
<head>
<title>Titel der Datei</title>
<style type="text/css">
  body { background-image:url(dateiname.jpg); }
</style>
</head>
<body>
<h1>Überschrift</h1>
</body>
</html>
```

- Bildkachelung kann beliebig angegeben werden mit:
 - background-repeat
 - Mögliche Werte: repeat-x, repeat-y, no-repeat

Beispiele für CSS-Eigenschaften

Hintergrundfarbe & -bilder

- Bildposition festlegen: background-attachment
 - Mögliche Werte: fixed (Wasserzeicheneffekt), scroll

```
<style type="text/css">
body { background-image:url(dateiname.jpg);
      background-attachment: fixed;
    }
</style>
```

- Weitere Eigenschaften: background-position
 - Mögliche Werte: left, right, center

DIV zur freien Gestaltung

```
<div>
  Dies ist der Titel
</div>
<div>
  Menübereich
</div>
<div>
  Hauptinhalt
</div>
```

- DIV-Elemente definieren rechteckige Bereiche
- Ermöglichen freie Festlegung von Bereichen, beispielsweise für:
 - Titel
 - Menü
 - Hauptinhalt
 - Infoboxen

Boxmodell

- Grundlage für den Seitenaufbau
- Gilt nicht nur für DIV
- Festlegung von Breite, Höhe, Innen- und Außenabständen
- Prinzip `<div>`

Inhalt:
weitere div-Elemente,
Absätze: `<p>`, etc.

`</div>`

Boxmodell

Das Diagramm zeigt ein zentrales Rechteck 'Inhalt' mit den Dimensionen 'width' und 'height'. Um dieses herum sind vier Ebenen definiert: 'padding' (Innenabstand), 'border' (Rahmen) und 'margin' (Außenabstand). Die gesamte äußere Begrenzung ist als 'Außenabstand' bezeichnet.

Boxmodell, Abstände

Das Diagramm zeigt zwei nebeneinander liegende Boxen. Die Abstände sind von außen nach innen wie folgt beschriftet: margin-top, border-top, padding-top, inhalt, padding-bottom, border-bottom, margin-bottom. Entsprechend sind die vertikalen Abstände (margin-left, border-left, padding-left, inhalt, padding-right, border-right, margin-right) für die horizontale Ausdehnung beschriftet.

Gesamtbreite und -höhe ergibt sich aus der Summe aller Einzelwerte in x und y-Richtung

Boxmodell, Positionierung

```
<div id="title">
  Dies ist der Titel
</div>
<div id="menu">
  Menübereich
</div>
<div id="content">
  Hauptinhalt
</div>
```

```
#title{
width: 500px;
height: 100px;
border: 1px solid black;
text-align: center;
}
#menu{
width: 500px;
height: 30px;
border: 1px solid black;
text-align: center;
margin-top: 5px;
}
```

HTML und CSS

27

- HTML beschreibt Struktur und Inhalt der Daten
 - Tags für Struktur
 - Inhalt „normaler“ Text
 - Struktur beinhaltet auch Markierung der Teile, die anders formatiert werden sollen
- Formatierung wird in Stylesheets (CSS) festgelegt
 - für jedes Element kann einzeln Formatierung bestimmt werden
 - Browser-Stylesheets sorgen für Standard-Darstellung
 - Autoren-Stylesheets überschreiben diese Einstellungen und können für individuelle Formatierungen verwendet werden

Formate für verschachtelte Elemente

28

- Bisher: Festlegen von Formaten für HTML-Elemente ohne Berücksichtigung der Position innerhalb des HTML-Dokuments
- Weiter: Formatangaben unter Berücksichtigung der Verschachtelung von HTML-Elementen

```
<p>To be or not to be...</p>
<div>
  <p>Ein berühmtes Zitat aus der Tragödie Hamlet, ...</p>
</div>
```

- Wie kann man nur die <p>-Elemente formatieren, die innerhalb von <div> stehen?

Formate verschachtelt angeben

29

```
<style type="text/css">
  div p { color:red; }
  div > p { color:blue; }
  div * b { color:violet; }
  div + p { margin-top:100px; }
</style>
```

- wird in <div>-Sektion rot gesetzt
- Text wird in <div>-Sektion blau gesetzt, wenn es direkt verschachtelt ist, z.B. <div><p>text</p></div>
- Text wird in <div>-Sektion violett gesetzt, wenn es indirekt verschachtelt ist, z.B. <div><p>text2</p></div>
- Text wird mit 100 Pixel Abstand gesetzt, wenn er direkt nach <div>-Sektion auftritt

Formate attributabhängig angeben

30

- Rot, wenn h1 Attribut „title“ enthält
- Blau, wenn h1 Attribut „Kapitel“ mit der Zuweisung "uebersicht" enthält
- Rot, wenn Absatz p Attribut „align“ enthält
- Grün und linksbündige Textausrichtung, wenn align "center" zugewiesen wurde

```
<style type="text/css">
  h1[title] { color:red; }
  h1[kapitel="uebersicht"] { color:blue; }
  p[align] { color:red; }
  p[align=center] { color:green; text-align:left; }
</style>
```

Cascading Style Sheets - CSS

31

- Stylesheets für unterschiedliche Ausgabegeräte
- wichtig für z.B. Barrierefreiheit
- Definition in HTML:

```
<html>
  <head>
    <title>Titel der Datei</title>
    <link rel="stylesheet" media="screen" href="website.css">
    <link rel="stylesheet" media="print, embossed" href="druck.css">
    <link rel="stylesheet" media="aural" href="speaker.css">
  </head>
  <body>
    ...
  </body>
</html>
```

Cascading Style Sheets - CSS

32

- Stylesheets für unterschiedliche Ausgabegeräte
- Definition in CSS:

```
@media print {
  /*Formatdefinitionen zum Drucken*/
  selektor {eigenschaft:wert;}
}

@media screen {
  /*Formatdefinitionen für BildschirmAusgabe*/
  selektor {eigenschaft:wert;}
}
```

Medientypen

Werte für das media-Attribut und @media

- all: CSS gilt für alle Medientypen
- aural: für synthetische Sprachausgabe
- braille: für Braille-Ausgabegerät
- embossed: für Braille-Drucker
- handheld: für kleine Geräte, Organizer, Handy, etc.
- print: für den Ausdruck auf Papier
- projection: für Projektion, bspw. mit Beamern
- screen: für Bildschirmausgabe
- tty: für nicht-graphische Ausgabe mit fester Zeichenbreite, bspw. Fernschreiber oder textorientierte Browser wie lynx
- tv: für TV-ähnliche Ausgabegeräte, meist mit grober Bildschirmauflösung

<http://de.selfhtml.org/css/formate/einbinden.htm>

Cascading Style Sheets - CSS

- Stylesheets können (und werden) hintereinander gehängt
 - und überschreiben einander
- Browser Stylesheet wird durch Autoren Stylesheet überschrieben
- mehrere Autorenstylesheets überschreiben einander
- dadurch können sehr komplexe Layouts realisiert werden, aber auch komplex in der Anwendung

Cascading Stylesheets

Die Kaskade

- Stylesheets können drei Quellen haben:
 - Autor des HTML-Dokuments
 - Benutzer: Nutzer können Styleinformationen für ein spezielles Dokument festlegen
 - User-Agent: müssen ein Default-Stylesheet besitzen oder so tun als hätten sie eins.
 - ✘ Sollten allgemeine Erwartungen erfüllen → physikalische Formatierung
- Stylesheets überlappen und beeinflussen sich entsprechend der Kaskade

<http://www.w3.org/TR/CSS21/cascade.html#cascade>

User Stylesheet

- Firefox: userContent.css
 - profile-directory/chrome/
 - Beispiel in: Mozilla Firefox\defaults\profile\chrome\userContent-example.css
- Opera: unter Einstellungen → Inhalte → Darstellungsoptionen eigenes Stylesheet auswählen
- Internet Explorer: unter Extras → Internetoptionen → Allgemein Barrierefreiheit wählen und Benutzerstylesheet festlegen



Stylesheets auf mehrere Dateien verteilen

- Bisher: einbinden mittels link-Element → HTML-Syntax
- Innerhalb des Stylesheets können CSS-Dateien mit @import eingefügt werden → CSS-Syntax

```
<HEAD>
<STYLE>
  @import "formate.css";
  @import url("druck.css") print;
  @import url("minipc.css") handheld;
</STYLE>
</HEAD>
```

im HTML-Dokument

```
@import "formate.css";
@import url("druck.css") print;
@import url("minipc.css") handheld;
```

im separaten Stylesheet
Wichtig: muss am Anfang
Des Stylesheets stehen.

Ein etwas ausführlicheres Beispiel:

```
<body>
<div id="body">
<div id="logo"></div>
<div id="title"></div>
<div class="menu">
<div class="menuItem"><a href="index.html">Home</a></div>
<div class="menuItem"><a href="1.html">Lehre</a></div>
<div class="menuItem"><a href="a.html">Arbeit</a></div>
<div class="menuItem"><a href="p.html">Publikationen</a></div>
</div>
<div id="content">
<ul id="contact">
<li id="name">Dr.-Ing. Marcel Götz</li>
<li id="institut">Dehmat</li>
<li id="school">Fakultät für Geistes-, Sozial- & Erziehungswissenschaften</li>
<li id="university">Otto-von-Guericke Universität Magdeburg</li>
<li class="empty"></li>
<li id="address">Adresse:</li>
<li id="street">Zachokkenstrasse 32</li>
<li id="city">39104 Magdeburg</li>
<li id="tel">Tel: ++49(0)391 67 16491</li>
<li id="fax">Fax: ++49(0)391 67 16541</li>
<li id="email">email: marcel.goetze@ovgu.de</li>
</ul>
</div>
</body>
```

Hilfsmittel

- Firebug: Firefox-Addon, das:
 - die Arbeit mit HTML-Quellcode erleichtert
 - Die Anzeige und Manipulation von CSS-Elementen erlaubt
 - Dem Programmierer die Kontrolle der Abarbeitung von Scripten (bspw. Javascript) ermöglicht → wichtig für die Fehlersuche
- TopStyle-Lite: Programm, das die Arbeit mit CSS erleichtert
 - CSS – Parameter können aus Liste gewählt werden (verhindert Fehler bei der Eingabe)
- Web-Developer Toolbar: Firefox-Addon

Zusammenfassung

- HTML beschreibt Daten
- CSS beschreibt Layout
- Angaben über Aussehen eines HTML-Elementes können:
 1. dem Element selbst über `style`-Attribut mitgegeben werden
 2. innerhalb eines `<style></style>`-Elementes angegeben werden
 3. in einer separaten `css`-Datei stehen
- bei Punkt 2 und 3 muss Zuordnung zu Element(n) erfolgen:
 - für alle HTML-Elemente eines Typs: Angabe des Tag-Namens im Stylesheet: bspw. `h1`
 - über `class`-Attribut des Elementes: → Selektor im Stylesheet fängt mit Punkt an: `.name`
 - über `id`-Attribut des Elementes: → Selektor im Stylesheet fängt mit Raute an: `#name`
- freie Gestaltungsmöglichkeiten über `div`-Element
- Basis für das Layout: Boxmodell