Kapitel 10: Cascading Style Sheets

Literatur:

- Erik Wilde: World Wide Web — Technische Grundlagen. Springer, 1999, ISBN 3-540-64700-7, 641 Seiten.
- Eric Ladd, Jim O'Donnell, et al.: Using HTML 4, XML, and Java 1.2, Platinum Edition. QUE, 1999, ISBN 0-7897-1759-X, 1282 pages.
- W3C: Web Style Sheets Home Page. `http://www.w3.org/Style/`
- W3C: Cascading Style Sheets Home Page. `http://www.w3.org/Style/CSS/`
- Håkon Wium Lie, Bert Bos: Cascading Style Sheets, level 1. W3C Recommendation. [http://www.w3.org/TR/REC-CSS1]
- Bert Bos, Håkon Lie, Chris Lilley, Ian Jacobs: Cascading Style Sheets, level 2. W3C Recommendation. [http://www.w3.org/TR/REC-CSS2/]
- Håkon Wium Lie, Bert Bos: Cascading Style Sheets, Designing for the Web. Addison Wesley, 2nd Edition, 1999, ISBN 0-201-59625-3, 413 pages.
- Dave Raggett, Arnaud Le Hors, Ian Jacobs (Eds.): HTML 4.01 Specification. W3C, 24.12.1999. [http://www.w3.org/TR/html4/]
- Stefan Münz: HTML-Dateien selbst erstellen — SELFHTML. [http://www.netzwelt.com/selfhtml/]
- Wikipedia: [http://en.wikipedia.org/wiki/Cascading_Style_Sheets]

Lernziele

- Verständnis für den Nutzen von Style Sheets.
- Eigene Entwicklung von Style Sheets, zumindest in relativ einfachen Fällen.
- Übersicht über Literatur/Quellen für weitere Informationen.

Inhalt

Motivation (1)

- Dokumente im WWW können auf unterschiedliche Arten verwendet werden:
  - ◇ Darstellung im Browser-Fenster.
  - ◇ Darstellung in kleinem Display (PDA).
  - ◇ Ausdruck auf Papier.
  - ◇ Ausgabe in Blindenschrift, Sprachausgabe.
  - ◇ Eintragung in Suchmaschinen.
  - ◇ Verarbeitung durch Programme (Shopbots, etc.)
- Jede Anwendung benötigt unterschiedliche Angaben zur Darstellung.

Motivation (2)

- HTML hatte anfangs vor allem inhalts-orientierte Elemente.
- Dann führten aber die Browser-Hersteller viele neue Elemente und Attribute ein, die das Aussehen der Seiten im Browser-Fenster beschreiben.

  Dies hatte auch den Zweck, Benutzer an einen Browser zu binden. Auf den Seiten stand oft „`Best viewed with ...`“. Es hatte auch etwas damit zu tun, daß das Web zuerst für Wissenschaftler entwickelt wurde, aber dann auch von Firmen verwendet wurde.
- Das Markup wurde also immer mehr darstellungs-orientiert.

Motivation (3)

- So wurde HTML immer umfangreicher, undurchschaubarer, und schwieriger für andere Medien zu verarbeiten.

  HTML war so auch einer schnellen Änderung unterworfen.
- Deswegen wurde bei HTML 4 eine Trennung eingeführt:
  - ◇ HTML selbst sollte nur den Inhalt beschreiben,
  - ◇ das Aussehen sollte mit Hilfe von Style Sheets festgelegt werden.

Motivation (4)

- Die Angaben zum Aussehen sind häufig willkürlich (Geschmackssache). Auch deswegen ist eine Trennung vom eigentlichen Inhalt wünschenswert.
- Für Anwendungen wie Suchmaschinen sind Angaben zum Aussehen irrelevant.
- Die Einführung besserer Gestaltungsmöglichkeiten war auch nötig, weil HTML-Elemente wie Tabellen und `BLOCKQUOTE` mißbraucht wurden, und viel Text als Bild in die Seiten eingefügt wurde.

Motivation (5)

- Für andere Medien (Ausdrucke, Sprachausgaben) sind auch andere Festlegungen zum Aussehen notwendig, als für einen Web-Browser.
- Eventuell möchten verschiedene Benutzer den Text auf verschiedene Arten angezeigt bekommen.
- Es kann daher zu einer HTML-Datei verschiedene Style Sheets geben.
- Man kann auch umgekehrt in vielen HTML-Dateien das gleiche Style Sheet verwenden.

  Alle Seiten einer Webpräsenz (“Site”) sollten einheitlich aussehen.

Cascading Style Sheets (1)

- HTML ist nicht an eine bestimmte Sprache für Style Sheets gebunden.

  Man muß daher in HTML deklarieren, daß man "Cascading Style Sheets" (CSS) verwendet.

- CSS ist nicht nur für HTML geeignet, sondern auch für XML.

  Man kann sich fragen, warum das W3C zwei Style Sheet Ansätze standardisiert hat (CSS und XSL). XSLT hat aber eine ganz andere Aufgabe als CSS (Transformation, Umsortierung/Umstrukturierung von Elementen). Insbesondere kann man mit XSLT auch in HTML+CSS übersetzen. CSS und XSL FO basieren auf dem gleichen zugrundeliegenden Formatierungs-Modell.

Cascading Style Sheets (2)

- Das Wort "Cascading" drückt aus, daß man mehrere Style Sheets kombinieren/vermischen kann (d.h. Angaben verschiedener Prioritäten haben).

  Zum Beispiel könnte die Firma ein Style Sheet für ihre „Corporate Identity" vorgeben, das eventuell für jede Abteilung leicht modifiziert werden kann, anschließend für jedes Dokument, und dann kann es noch Ausnahmen in den einzelnen Elementen im Text geben.

- Insbesondere kann auch der Leser der Dokumente (Benutzer des Browsers) Wünsche für die Darstellung äußern, nicht nur der Autor des Dokumentes.

  Dies führte bei der Vorstellung des Vorschlags auf der zweiten WWW-Konferenz zu Diskussionen über Rechte von Autoren und Lesern.

Cascading Style Sheets (3)

- Mit CSS1 kann man z.B. definieren:
  - ◇ Schriftart, Schriftgröße
  - ◇ Farben
  - ◇ Einrückungen, horizontaler/vertikaler Leerplatz
  - ◇ Begrenzungslinien, Ränder
  - ◇ Ausrichtung, z.B. Zentrierung
- Insgesamt bekommt man mehr Möglichkeiten, als HTML jemals an darstellungs-orientierten Elementen und Attributen hatte.

Cascading Style Sheets (4)

- Man kann mit Style Sheets auch das übliche Verhalten der Browser definieren („Default Style Sheet").

  Im CSS1 Standard ist ein Default Style Sheet für HTML 2.0 angegeben. Im CSS2 Standard gibt es eine für HTML 4. Das verhalten einiger weniger Elemente kann aber noch nicht in CSS 2 definiert werden (z.B. `img`, Applets, Frames, Formulare).

- Die Darstellung der HTML-Elemente ist dann nicht mehr fest in den Browser eingebaut, sondern über Daten definiert.

  Tatsächlich war schon der erste Browser von Tim Berners-Lee so aufgebaut, daß die Darstellung der Elemente über ein Style Sheet gesteuert werden konnte. Die Style Sheet Sprache wurde damals aber nicht veröffentlicht.

Geschichte (1)

- Ein erster Entwurf für "Cascading HTML Style Sheets" von Håkon Wium Lie erschien 1994.

  Der Vorschlag wurde auf der WWW Konferenz im November 1994 vorgestellt. Er wurde kombiniert mit Ideen von Bert Bos, der eine Style Sheet Sprache in seinem Browser "Argo" implementiert hatte.

- Cascading Style Sheets, level 1 (CSS1) wurde vom W3C am 17.12.1996 verabschiedet.

  Eine überarbeitete Version wurde am 11.01.1999 veröffentlicht.

- Cascading Style Sheets, level 2 (CSS2) wurde am 12.05.1998 verabschiedet.

Geschichte (2)

- CSS 2.1 ("CSS Level 2 Revision 1") ist seit dem 19.07.2007 eine "Candidate Recommendation".

  Es enthält die Features, die zum Zeitpunkt der Veröffentlichung allgemein implementiert sind. Damit werden auch einige Features aus CSS2 weggelassen (auf CSS3 verschoben). Es gibt aber auch einige neue Möglichkeiten.

  Tatsächlich war CSS 2.1 schon einmal eine Cadidate Recommendation (seit 25.02.2004), aber wurde am 13.06.2005 in den "Working Draft" Status zurückgestuft.

- Das W3C arbeitet an CSS3.

  Dies scheint wesentlich größer zu sein und wird auf mehrere Module verteilt sein.

Geschichte (3)

- Der erste Browser, der einen Teil von CSS unterstützte, war Internet Explorer 3 (August 1996).

  Es fehlten große Teile vom "Box Model".

- Netscape Navigator 4 unterstützte im Prinzip einen großen Teil von CSS1, aber die CSS Implementierung enthielt noch viele Fehler.

  Netscape 3.x verstand noch keine Style Sheets.

- Auch Internet Explorer 4 unterstützte relativ viel von CSS1, aber mit Fehlern.

Geschichte (4)

- Opera 3.5 (Nov. 1998) unterstützte den größten Teil von CSS1.
- Internet Explorer 5.0 für den Mac (März 2000) war der erste Browser, der CSS1 fast vollständig unterstützte (mehr als 99%).
- Bis heute (Juli 2008) gibt es keinen Browser, der CSS2 vollständig unterstützt.
- Die vielen Fehler und Inkompatibilitäten waren ein ernstes Hindernis in der Verbreitung von CSS.

Inhalt

Einbindung in HTML (1)

- Style Sheet Regeln können ein drei Stellen stehen:
  - ◇ In einer vom HTML-Dokument getrennten Datei, auf die mit `link` verwiesen wird.

    Ein Style Sheet kann auch mit dem HTTP-Header `Default-Style` (oder dem entsprechenden `meta`-Element) festgesetzt werden.
  - ◇ Innerhalb des Elementes `style` im Dokument-Kopf.
  - ◇ Jeweils in dem betroffenen Element im Attribut `style`.
- Die Vor- und Nachteile werden auf den folgenden Folien erläutert.

Einbindung in HTML (2)

Style Sheet in eigener Datei:

- Vorteile:
  - ◇ Das gleiche Style Sheet kann für mehrere Seiten verwendet werden.
  - ◇ Nur benötigte Stylesheets werden geladen.

    Wenn es mehrere alternative Style Sheets gibt.
- Nachteile:
  - ◇ Der Browser muß das Style Sheet extra anfordern und kann den Text erst danach darstellen.

    Wenn das gleiche Style Sheet in mehreren Seiten eingesetzt wird, betrifft dieses Problem nur die erste Seite (Caching im Browser).

Einbindung in HTML (3)

Style Sheet im Dokumentkopf:

- Vorteile:
  - ◇ Wird gleich mit dem Dokument mit geladen, die HTML Seite kann daher inkrementell aufgebaut werden (während die Daten noch kommen).
- Nachteile:
  - ◇ Keine Mehrfachverwendung von Stylesheets.

    Dadurch Redundanzen und Gefahr von Inkonsistenzen.
  - ◇ Verwirrt möglicherweise ganz alte Browser, die `style` nicht kennen.

Einbindung in HTML (4)

Angaben im `style`-Attribut bei betroffenen Elementen:

- Vorteile:
  - ◇ Einfach, direkt.

    Es steht direkt beim betroffenen Element, man braucht keinen Selektor anzugeben.
- Nachteil:
  - ◇ Man kann keine alternativen Angaben machen.

    Zum Beispiel für verschiedene Ausgabemedien.
  - ◇ Inkonsistenzen durch sehr lokale Angaben.

Einbindung in HTML (5)

Beispiel (Style Sheet in eigener Datei):

- Z.B.: `h1` Überschriften sollen in roter 14pt Schrift erscheinen.
- Man legt eine Datei, z.B. `my.css`, mit folgendem Inhalt an:

```
h1 { color: red; font-size: 14pt}
```

- Diese kann man nun mit einem `link`-Element referenzieren:

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


Einbindung in HTML (6)

Beispiel (Style Sheet im Kopf des Dokumentes):

- Hier steht das Style Sheet im Element `style`:

```
<style type="text/css">
    h1 { color: red; font-size: 14pt}
</style>
```

- Alte Browser, die `style` nicht kennen, nehmen an, daß hier bereits der `body` beginnt.
- Deswegen wird die Stylesheet-Information manchmal in einen HTML-Kommentar eingeschlossen:

```
<style type="text/css"><!--
    h1 { color: red; font-size: 14pt}
--></style>
```


Einbindung in HTML (7)

Style Sheet Angabe im Kopf, Forts.:

- Das Element `style` ist folgendermaßen deklariert:

```
<!ELEMENT style - -    CDATA>
<!ATTLIST style %i18n;
                type   %ContentType; #REQUIRED
                media  %MediaDesc;   #IMPLIED
                title  %Text;        #IMPLIED >
```

- Da der Inhalt als `CDATA` deklariert ist, wird Markup nicht ausgewertet (insbesondere keine Kommentare gelöscht).

  Kommentare in Stylesheets schreibt man wie in C: `/*...*/`.

Einbindung in HTML (8)

Beispiel (Style Sheet Angabe im Attribut):

- Alternativ kann man die Angabe auch direkt bei den betroffenen Überschriften machen:

```
<h1 style="color: red; font-size: 14pt">
```

- Das Attribut `style` ist Bestandteil von `%attrs;`. Es ist bei praktisch jedem Body-Element möglich.
- Der HTML Standard verlangt, daß die Style Sheet-Sprache immer deklariert ist. Oben geschieht dies mit dem Attribut `type`, hier mit einem Meta-Tag:

```
<meta http-equiv="Content-Style-Type"
      content="text/css">
```


Alternative Style Sheets (1)

- Soll das Style Sheet auch zum Ausdrucken verwendet werden, so setzte man in den Elementen `link` bzw. `style` noch folgendes Attribut:

```
media="screen,print"
```

- Der Default ist `media="screen"`.

  D.h. das Style Sheet wird zum Drucken nicht verwendet, sondern die Default-Setzungen des Browsers (oder alternatives Style Sheet).

- Bei der dritten Variante (`style`-Attribut) kann man keine alternativen Style Sheets angeben.

Alternative Style Sheets (2)

- Ausgabemedien:
  - ◇ `screen` (normaler Computer-Bildschirm),
  - ◇ `tty` (festes Raster von Zeichen),
  - ◇ `tv` (kleine Auflösung, kaum scrollen),
  - ◇ `projection` (Beamer),
  - ◇ `handheld` (kleiner Bildschirm, schwarzweiss, . . .),
  - ◇ `print` (eingeteilt in Seiten),
  - ◇ `braille` (Schrift zum Tasten)
  - ◇ `aural` (Sprachausgabe),
  - ◇ `all` (geeignet für alle Geräte).

Alternative Stylesheets (3)

- Es ist möglich, mehrere alternative Stylesheets im Dokument anzugeben, und dem Benutzer die Auswahl zu überlassen.

  Falls der Browser das unterstützt.

- Dazu hat man mehrere `link`-Elemente mit Verweisen auf Stylesheet-Dateien.
- Der vom Autor vorgeschlagene Default (“preferred stylesheet”) verwendet `rel="stylesheet"`, alle anderen verwenden

```
rel="alternate stylesheet"
```

Alternative Stylesheets (4)

- In diesem Fall (mehrere alternative Style Sheets) muß das `title`-Attribut aller dieser `link`-Elemente gesetzt werden (der Browser kann dann diese Namen dem Benutzer zur Auswahl anbieten).

  Style Sheets mit gleichem `title` sind möglich (immer zusammen).

- Wird für ein `link`-Element kein `title` angegeben, und ist `rel=stylesheet`, so zählt es als "persistent" und muß immer angewendet werden (ggf. zusätzlich zur Benutzerauswahl).

  Aber nur, wenn die `media`-Auswahl paßt. Außerdem können Browser erlauben, Stylesheets ganz abzuschalten.

Alternative Style Sheets (5)

- Zumindest bei CSS ist es möglich, Angaben aus mehreren Stylesheets zu mischen, z.B.
  - ◇ mehrere passende `link`-Elemente, oder
  - ◇ `link` und zusätzlich `style` etc.
- Die Prioritätsregeln sind relativ kompliziert (s.u.), aber wenn es sonst keinen Unterschied gibt, gewinnt die spätere Angabe.

Inhalt

Syntax (1)

- Abgesehen von `@import`-Anweisungen ist ein Style Sheet eine Menge von Regeln.
- Eine Regel besteht aus
  - ◇ einem "Selector" (zur Auswahl der betroffenen Elemente) und
  - ◇ einer "Declaration" (Festlegung von Eigenschaften der Darstellung) in geschweiften Klammern.

$$\underbrace{\texttt{h1}}_{\text{Selector}} \quad \{\ \underbrace{\texttt{color: green}}_{\text{Declaration}}\ \}$$

Syntax (2)

- Eine Deklaration besteht wiederum aus zwei Teilen (getrennt durch einen Doppelpunkt):
  - ◇ einer Eigenschaft ("Property") und
  - ◇ einem Wert ("Value").

$$\underbrace{\texttt{h1}}_{\text{Selector}}\ \texttt{\{}\ \underbrace{\texttt{color}}_{\text{Property}}\ \texttt{:}\ \underbrace{\texttt{green}}_{\text{Value}}\ \texttt{\}}$$

- Ein Wert ist nicht notwendigerweise ein einzelnes Wort, z.B. ist auch folgendes möglich:

```
body { background: url(texture.gif) white }
```

Wenn das Bild nicht verfügbar ist, wird der Hintergrund weiß gemacht.

Syntax (3)

- Im "Declaration"-Teil können mehrere Eigenschaften gesetzt werden. Sie müssen durch Semikolon getrennt werden. Ein Semikolon am Ende ist erlaubt (optional):

```
h1 { font-weight: bold; color: green }
```

- Man kann Leerplatz, Zeilenumbrüche, Kommentare beliebig einfügen, z.B.

```
h1 { /* Dies ist ein Kommentar */
    font-weight: bold;
    color: green;
}
```


Syntax (4)

- Mehrere Regeln mit der gleichen Eigenschaftsliste können zusammengefasst werden, dazu werden die Selektoren durch Kommata getrennt:

```
h1, h2 { color: green }
```

ist äquivalent zu

```
h1 { color: green }
h2 { color: green }
```


Auswahl von Elementen (1)

- Oben waren die Selektoren zur Auswahl der betroffenen Elemente immer Element-Typen, z.B.

```
h1 { color: red}
```

- Wenn das Stylesheet für HTML verwendet wird, ist die Groß-/Kleinschreibung dabei egal.
- Man kann eine bestimmte Schachtelung verlangen, z.B. `li`-Elemente innerhalb eines `ul`-Elementes:

```
ul li { color: red; font-size: 14pt}
```

Auswahl von Elementen (2)

- Man kann aber z.B. auch Elemente im HTML Quelltext mit dem Attribut `class` markieren,

```
<h1 class="wichtig">Kapitel 1</h1>
```

- Im Stylesheet sind die Klassen-Angaben durch einen vorangestellten Punkt gekennzeichnet:

```
.wichtig { color: red; font-size: 14pt}
```

- Man kann auch Typ und Klasse kombinieren:

```
h1.wichtig { color: red; font-size: 14pt}
```


Auswahl von Elementen (3)

- Entsprechend kann man ein bestimmtes Element mit einer ID markieren:

```
<h1 id="kap1">Kapitel 1</h1>
```

- IDs sind im Stylesheet mit dem vorangestellten Symbol "`#`" markiert:

```
#kap1 { color: red; font-size: 14pt}
```

- Bei Links gibt es auch "pseudo classes" zur Klassifizierung der Links: `visited` (schon besucht), `link` (noch nicht besucht), `hover` (Maus darüber)

```
a:visited { color: red}
```


Auswahl von Elementen (4)

- Fast alle Eigenschaften vererben sich im Baum, d.h. gelten auch für geschachtelte Elemente, sofern sie nicht überschrieben werden.
- Setzungen für `body` gelten also für das ganze Dokument, wenn nicht explizit eine andere Auswahl getroffen wird.

Inhalt

Zeichensatz Auswahl

- `font-family`,
- `font-style`,
- `font-variant`,
- `font-weight`,
- `font-size`,
- `font`.

Farben, Hintergrund

- `color`,
- `background-color`,
- `background-image`,
- `background-repeat`,
- `background-attachment`,
- `background-position`,
- `background`.

Text Eigenschaften

- `word-spacing`,
- `letter-spacing`,
- `text-decoration`,
- `vertical-align`,
- `text-transform`,
- `text-align`,
- `text-indent`,
- `line-height`.

“Box” Eigenschaften (1)

- `margin-top`, `margin-right`, `margin-bottom`, `margin-left`, `margin`,
- `padding-top`, `padding-right`, `padding-bottom`, `padding-left`, `padding`,

"Box" Eigenschaften (2)

- `border-top-width`, `border-right-width`,
  `border-bottom-width`, `border-left-width`,
  `border-width`,
  `border-color`,
  `border-style`,
  `border-top`, `border-right`, `border-bottom`,
  `border-left`, `border`,

"Box" Eigenschaften (3)

- `width`,
- `height`,
- `float`,
- `clear`.

Klassifizierungs Eigenschaften

- `display`,
- `white-space`,
- `line-style-type`,
- `line-style-image`,
- `line-style-position`,
- `line-style`.