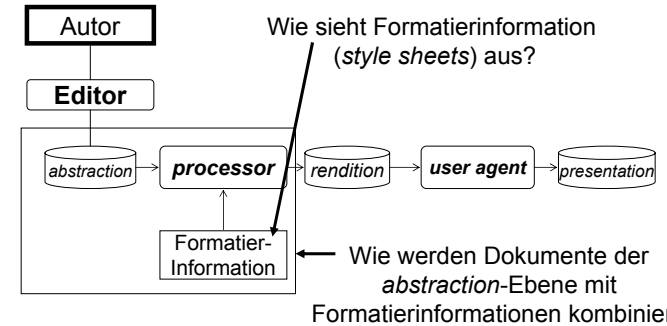


Style Sheets (Teil 1)

Einbettung von Style Sheets
CSS2 - Eine kleine Einführung

(Erinnerung)

Offene Fragen:



Einbettung von Style Sheets

W3C Standard:
„Associating Style Sheets with
XML documents“ (Version 1.0)

- Der Kontext
 - Referenzieren von Darstellungsinformation (style sheets)
- Die Aufgabe
 - Festlegung einer einfachen (Interims-) Norm, um für ein vorliegendes XML-Dokument passende style sheets (CSS, XSL) direkt aus dem XML-Quelltext heraus anzusprechen.
- Der Weg
 - Implementierung über processing instructions (PI)
 - Syntax eng an Attributlisten von XML-Elementen angelehnt (Pseudoattribute)
 - PI soll genauso wie ein start tag ausgewertet werden.
 - Vorbild HTML 4.0-Element „LINK“

style sheets referenzieren

- Einschränkungen und Anmerkungen
 - Es dürfen mehrere derartige PIs vorkommen
 - Diese PIs dürfen nur im Prolog des XML-Dokuments vorkommen
 - Bitte die nähere Bedeutung der Pseudoattribute in der analogen Dokumentation zu HTML 4.0 nachschlagen.
 - Seltsame Anmerkung im Standard:
„This replacement [of a CharRef or a PredefEntityRef] by its character is not performed automatically by an XML processor.“

28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

5

style sheets referenzieren

- Beispiele und entsprechende HTML-Versionen

```
<LINK href="mystyle.css" rel="style sheet"
type="text/css">
<?xml-stylesheet href="mystyle.css"
type="text/css"?>

<LINK href="mystyle.css" title="Compact"
rel="stylesheet" type="text/css">
<?xml-stylesheet href="mystyle.css"
title="Compact" type="text/css"?>

<LINK href="mystyle.css" title="Medium"
rel="alternate stylesheet" type="text/css">
<?xml-stylesheet alternate="yes"
href="mystyle.css" title="Medium"
type="text/css"?>
```

28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

6

style sheets referenzieren

- Die Grammatikregeln

[1] StyleSheetPI ::=

```
'<?xml-stylesheet' (S PseudoAtt)* S?'?>'
```

[2] PseudoAtt ::= Name S? '=' S? PseudoAttValue

[3] PseudoAttValue ::=

```
('"' ([^"<&] | CharRef | PredefEntityRef)* '"' |
'"'"' ([^'<&] | CharRef | PredefEntityRef)* "'"' ) -
(Char* '?>' Char*)
```

[4] PredefEntityRef ::= '&#' | '<#' | '>#' | '"#' | ''#'

28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

7

style sheets referenzieren

- Unterstützte Pseudoattribute:

href	CDATA	#REQUIRED	URI der style sheet - Quelle
type	CDATA	#REQUIRED	Typ.: "text/css" oder "text/xsl"
title	CDATA	#IMPLIED	
media	CDATA	#IMPLIED	
charset	CDATA	#IMPLIED	
alternate	(yes no)	"no"	Siehe Bemerkung 1
- Bemerkungen:
 1. Falls alternate="yes", entspricht dies in HTML: REL="alternate stylesheet" statt REL="stylesheet"
 2. Die hier verwendete Syntax zur Deklaration der Pseudoattribute ist gleich der bei XML-Attributen verwendeten. Zur Definition siehe dort (DTD, „ATTLIST“-Deklaration).
 3. Hintergrundinfo zu den hier ungenutzten, optionalen Attributen: Das HTML-Element LINK war für eine Vielfalt von Anwendungsfällen konzipiert worden. In der Praxis setzten sich dagegen nur wenige Fälle durch, entsprechend entfiel der Bedarf für einige Attribute.

28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

8

CSS2 - Eine kleine Einführung

Cascading Style Sheets Level 2 mit Schwerpunkt auf XML

(Einzelheiten: Siehe Originalspezifikationen!)

CSS2: Vorgeschichte

- HTML
 - HTML entstand ohne klare Trennung zwischen *abstraction* und *rendition*, aber mit Schwerpunkt auf der *abstraction*-Seite.
 - Die stürmische Entwicklung des WWW außerhalb des akademischen Ursprungs rückte rasch die Frage in den Vordergrund, wie Inhalte dargestellt werden.
 - Der so motivierte Bedarf nach HTML-Erweiterungen fachte einen „Browser-Krieg“ (insb. zwischen Netscape und Microsoft) an. Beide Browser-Hersteller entwickelten proprietäre, zueinander inkompatible HTML-Erweiterungen.
 - Das W3C reagierte mit der Spezifikation der CSS (Level 1).

CSS2: Vorgeschichte

- HTML (Forts.)
 - CSS sollte idealerweise komplett die *rendition*-Ebene übernehmen, HTML-Code sollte sich auf die *abstraction*-Ebene konzentrieren. Dem sind allerdings Grenzen gesetzt durch
 - inzwischen standardisierte HTML-Erweiterungen (bis HTML 4.0)
 - die konzeptionell unklare Trennung von *abstraction* und *rendition* in HTML
 - die Beschränkungen von HTML auf *abstraction*-Ebene, welche letztlich erst von XML überwunden werden.

CSS2: Vorgeschichte

- CSS
 - Der erste - noch relativ einfache - CSS-Level fand lange Zeit nur geringe bis mäßige Unterstützung durch die Browser-Hersteller.
 - Web-Entwickler, die zur Vermeidung von Browser-Inkompatibilitäten CSS einsetzten, gerieten so vom Regen in die Traufe.
 - Dies schadete dem Ruf von CSS - trotz seines bestechenden Konzepts - und erschwerte seine Verbreitung.
 - Mit CSS2 wurde bereits 1998 ein erheblich komplexerer Nachfolger als Standard spezifiziert, der weitgehend - aber nicht vollkommen - abwärtskompatibel zu CSS1 ist. Zu dieser Zeit war selbst CSS1 noch nicht hinreichend verbreitet.

CSS2: Vorgeschichte

- CSS (Forts.)
 - Inzwischen (2003) hat sich die Situation deutlich verbessert:
 - Aktuelle Browser mit XML-Unterstützung unterstützen nun auch CSS1 und die wichtigsten CSS2-Eigenschaften
 - Wer client-seitig ohnehin XML-Fähigkeiten voraussetzt, kann nun CSS2 einsetzen, ohne auf größere Probleme zu stoßen.
 - Allerdings sollten Entwickler von derartigen Website auch heute noch ihre Ergebnisse mit verschiedenen Browser (insb. IE, Netscape/Mozilla, Opera) testen!

28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

13

CSS2: Vorgeschichte

CSS2 und XML - das „dream team“ ?

- Konzeptionell passt CSS ideal zu XML:
 - XML bewegt sich rein auf der *abstraction*-Ebene,
 - CSS komplett auf der *rendition*-Ebene.
 - XML in Reinform kann von XML-fähigen Browsern nur in sehr generischer - und damit oft unbrauchbarer - Form angezeigt werden.
 - CSS bietet genau die Möglichkeiten, einem Browser mitzuteilen, wie XML-Daten anzuzeigen sind.
 - Dank des gemeinsamen SGML-Ursprungs läßt sich CSS auch direkt auf XML-Daten anwenden.
 - Seit CSS2 lassen sich sogar *style sheets* für verschiedene Medientypen - und deren spezifische Ansprüche - pflegen.

28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

14

CSS2: Vorgeschichte

CSS2 und XML - das „dream team“ ?

- Grenzen des Verfahrens:
 - Mit CSS2 können Sie festlegen, wie XML-Inhalte angezeigt werden sollten, aber nur bedingt welche.
 - Erfordert die XML-Struktur „Umbauten“ (Transformationen), bevor eine Anzeige sinnvoll ist, wird XSL(T) verwendet.

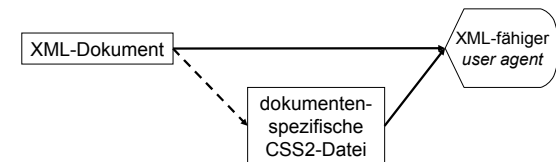
28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

15

CSS2: Einfacher Einsatz mit XML

- XML / CSS2
 - In einfachen Fällen genügt es, wenn XML-Quelldaten direkt auf CSS-Dateien verweisen.
 - Diese CSS-Dateien müssen allerdings auf den jeweiligen Dokumententyp abgestimmt sein.
 - Voraussetzung ist ferner die Verwendung eines modernen, XML-fähigen *Browsers (user agent)* auf *client*-Seite.



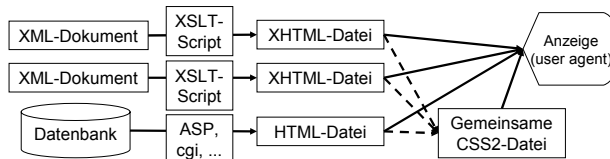
28.05.2003

H. Werntges, FB Informatik, FH Wiesbaden

16

CSS2: Aktuelles Einsatz-Szenario

- XML / XSLT / XHTML / CSS2
 - In komplexeren Website-Szenarien wandelt man XML-Quelldaten zunächst mit XSLT in XHTML-Dokumente um.
 - *rendition*-Details wie Fontgrößen, -arten, Farben, Liniendicken usw. dieser XHTML-Dokumente werden in eine CSS-Datei ausgelagert.
 - Dies entlastet die Transformation und verschlankt die XHTML-Dateien, beschleunigt also spätere Ladezeiten.
 - Wichtiger noch ist aber die Möglichkeit, einen gemeinsamen Stil über alle Teile einer Website einzuhalten, zentral zu pflegen, und bei Bedarf mit geringem Aufwand - an nur einer Stelle - abzuändern.



CSS2: Ihre Erwartungen...

Stichwortsammlung an der Tafel:

- Wie könnte man XML-Elemente zur Anzeige bringen?
- Welche Anzeige-Eigenschaften von XML-Elementen möchten Sie gerne beeinflussen können?

CSS2: Ein XML-Beispiel

- XML-Fragment (aus: Tutorial der CSS2-Spezifik., Kap. 2.2):

```
<?xml-stylesheet type="text/css" href="bach.css"?>
<ARTICLE>
<HEADLINE>Fredrick the Great meets Bach</HEADLINE>
<AUTHOR>Johann Nikolaus Forkel</AUTHOR>
<PARA>
  One evening, just as he was getting his
  <INSTRUMENT>flute</INSTRUMENT> ready and his
  musicians were assembled, an officer brought him a
  list of the strangers who had arrived.
</PARA>
</ARTICLE>
```

- Ein passendes CSS-Dokument dazu:

```
INSTRUMENT          { display: inline }
ARTICLE, HEADLINE, AUTHOR, PARA { display: block }
HEADLINE            { font-size: 1.3em }
AUTHOR              { font-style: italic }
ARTICLE, HEADLINE, AUTHOR, PARA { margin: 0.5em }
```

CSS2-Demo

- bach0 - XML ohne CSS:
 - Browserspezifische Darstellung, bei IE baumartig.
- bach1 - XML mit leerer CSS-Datei:
 - Nur die unformatierten Nutzdaten, „*inline style*“
- bach2 - XML mit CSS-Datei gemäß Beispiel:
 - Brauchbar!
 - Unterschiede im Detail zur Wiedergabe in den Spezifikationen
- bach3 - Variationen in CSS:
 - Hintergrundfarbe
 - Font: Helvetica, sans-serif
 - Breite fest vorgegeben, Titel und Autor zentriert
 - 3 Textattribute für „Instrument“.
- Beachte Browser-Unterschiede Mozilla-IE!

CSS2: Ein XML-Beispiel

- Die Anzeige der XML-Datei im Browser:
 - Schon recht brauchbar!

Fredrick the Great meets Bach

Johann Nikolaus Forkel

One evening, just as he was getting his flute ready and his musicians were assembled, an officer brought him a list of the strangers who had arrived.

CSS2: Aufbau

- Ein CSS *style sheet* besteht aus einer Folge von *statements*:
- Es gibt zwei Arten von *statements* (Anweisungen):
 - *at-rules* (@-Regeln)
 - *rule sets* bzw. *rules* (Regelmengen, Regeln)
- *at-rules*
 - Grammatik:

```
at-rule ::= '@' identifier S* ( [^;] ';' | block )
```
 - Beispiele:

```
@import "subs.css"
@media print { BODY { font-size: 10pt } }
```
- *rule sets / rules*
 - Grammatik:

```
rule ::= selector block
```
 - Beispiel:

```
H1, H2 {color: green}
```

CSS2: selector

- *selector*
 - Im einfachsten Fall der Elementname, dem bestimmte Eigenschaften im folgenden *declaration block* zugewiesen werden sollen.
- Grammatik

```
selector ::= ( type_selector | universal_selector )
           ( attribute_selector | ID selector | pseudo_class )*
```
- Beispiele
 - * passt zu jedem Element
 - E wählt jedes Element E aus
 - E F wählt F aus, wenn es von einem E abstammt (*descendant*)
 - E > F ..., wenn F ein direktes Unterelement (*child*) von E ist
 - E + F ..., wenn F direkt auf ein Element E folgte
 - E[foo] wählt E aus, wenn es ein gesetztes Attribut *foo* besitzt
 - E[foo="val"] ..., wenn sein Attribut *foo* den Wert "val" besitzt
 - E:first-child ..., wenn es das erste *child element* seines *parent* ist

CSS2: selector

- *selector grouping*
 - Mehreren *selectors* kann auf einfache Weise derselbe *declaration block* zugewiesen werden. Dazu liste man sie einfach komma-separiert auf.
 - Beispiel: **H1, H2, H3, P { ... }**
- Bemerkungen
 - Es gibt noch zahlreiche Details allein zu *selectors* zu beachten.
 - Das Thema CSS2 soll hier nur angedeutet und keineswegs erschöpfend behandelt werden, daher:
 - Einzelheiten siehe Kapitel 5 („Selectors“) der CSS2-Spezifikationen.

CSS2: *declaration blocks*

- (mündliche Kommentare)
- (Basiswissen erschließt sich leicht aus den Beispielen)
- (ggf. in den Spezifikationen nachlesen)

CSS2: „*cascading*“, Vererbung

- Vererbung:
 - „Normalerweise“ gemäß Documentenbaum-Struktur
 - Jede Eigenschaft definiert, ob sie vererbt wird oder nicht
 - Eigenschaftswert „inherit“. Beispiel: CSS2-6.2.1
- Cascading:
 - Drei Quellen: *default, user, author stylesheet*
 - Eigenschaften: Normal vs. !important
 - Importierte Daten
- (mündliche Kommentare; Basiswissen erschließt sich leicht aus den Beispielen)
- (ggf. in den Spezifikationen nachlesen)

CSS2: *media types, media groups*

- Mittels der in CSS2 aufgenommenen *at-rule* @media ist es nun möglich, Regeln für verschiedene Medien parallel und sauber getrennt in einer CSS-Datei zu pflegen.
- *media types*
 - *aural, braille, emboss, handheld, print, projection, screen, tty, tv*
- *media groups*
 - *continuous/paged, visual/aural/tactile, grid/bitmap, interactive/static*
- Matrix der Zuordnungen zwischen *media types* und *media groups*
 - Siehe 7.3.1
- Beispiel (aus: 7.2.1):

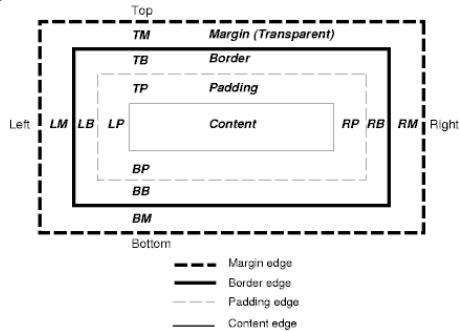
```
@media print { BODY { font-size: 10pt } }  
@media screen { BODY { font-size: 12pt } }  
@media screen, print { BODY { line-height: 1.2 } }
```

CSS2: *boxes, „display“ property*

- Allgemeines zu *boxes*
 - Typensatz: Lettern zu Zeilen / kleinen oder größeren Rechtecken
 - ...
- *Box-Arten*
 - *block-level elements, block boxes*
 - *block, list item*
 - *compact, run-in* (kontextabhängig)
 - *in-line level elements, in-line boxes*
 - *inline, inline-table*
 - *compact, run-in* (kontextabhängig)
 - Spezialfälle
 - *marker*: Erzeugt Nummern u.a. Randzeichen neben *block-level elements*
 - *none*: Erzeugt keine *box* für diesen *selector*.
Praktisch z.B. zum Ausblenden ungewünschter XML-Elemente!
 - Die **table**-Familie - siehe Seite „CSS2: Tabellen und XML“

CSS2: Box-Modell

- Elemente des Dokumentenbaums werden in rechteckige Kästen (boxes) umgewandelt
- Jeder Kasten besitzt einen Inhalt (content) sowie optionale Umgebung wie folgt:
 - Polsterung (padding),
 - Rahmen (border),
 - Randbereiche (margin)



CSS2 vs. frames und/oder Tabellen

- HTML-Gestaltung verwendet oft *frames* oder Tabellen zur Layoutkontrolle
- Beide Ansätze haben Nachteile!
- CSS2 bietet eine Alternative: Absolute Positionierung von Elementen / Boxen!
- Zur Diskussion von Vor- und Nachteilen:
 - Siehe auch: S. Mintert, CSS-Tutorial, Teil 2: Frames, Tabellen und XML. iX 4/2003, pp. 138-143

CSS2: Tabellen und XML

- Anleihe bei HTML-Tabellen:
 - HTML besitzt standardisierte Elemente zur Erzeugung von Tabellen
 - CSS2 besitzt dazu passende Strukturen zur Kontrolle des Aussehens von Tabellen, incl. Über/Unterschriften, Kopf- und Fußzeilen, Rahmen.
 - XML kann von dieser Infrastruktur in ebenso einfacher wie wirkungsvoller Weise Gebrauch machen - durch Assoziation geeigneter XML-Elemente mit funktional äquivalenten HTML-Elementen zum Tabellenaufbau.

CSS2: Tabellen und XML

- Entsprechende CSS display properties (in Klammern: HTML Element):
 - table*, *inline-table* (TABLE),
 - table-row* (TR), *table-row-group* (TBODY),
 - table-header-group* (THEAD), *table-footer-group* (TFOOT),
 - table-column* (COL), *table-column-group* (COLGROUP),
 - table-cell* (TD, TH),
 - table-caption* (CAPTION)
- Zuordnungsbeispiel

```
HBOX { display: table-row }
VBOX { display: table-cell }
```

(Annahme: XML-Dokument enthält Elemente HBOX, VBOX)

CSS2: Fehlerbehandlung

- **Kein Abbruch!**
 - Im Gegensatz zu XML herrscht bei CSS die „tolerante“ HTML-Tradition vor.
 - Grundregel:
 - Teile einer CSS-Datei, die syntaktisch nicht korrekt sind, werden ignoriert.
 - Gleiches trifft auf Fehler zu, deren Ursache in unbekanntem Eigenschaften bzw. Schlüsselwörtern haben.
 - Werden bestimmte Reihenfolgen nicht eingehalten, ignoriert der Browser (eigentlich: *User Agent*, UA) auch derartig falsch platzierte Anweisungen.
 - Beispiel: Ein `@include` erst *nach* Angabe des ersten *ruleset*
- Einzelheiten:
 - Siehe Abschnitt 4.2 der CSS2-Spezifikationen

CSS2: Prinzipielle Grenzen

- Darstellung von Verweisen
 - Referenzen:
 - `<ref idref="id-von-Kap-3">` sollte am besten mit einem Text (etwa: Überschrift) aus Kap.3 dargestellt werden. CSS bietet derartige Möglichkeiten nicht.
 - Bilder und Hyperlinks:
 - XML-Hyperlinks bleiben inaktiv
 - XML-Links auf Bilder bewirken keine Einbindung
- Ausweg: XLink
 - Für derartige Wünsche ist XLink zuständig, nicht CSS
 - XML-fähige Browser sollten daher auch XLink unterstützen!

CSS2: Nachwort

- Über den Umgang mit CSS2
 - Das hier präsentierte Material soll nur einen ersten Eindruck von den Möglichkeiten von CSS2 verschaffen.
 - Wer CSS ernsthaft einsetzen will, sollte sich mit den Spezifikationen selbst beschäftigen.
 - Diese sind umfangreich (19 Kapitel und 8 Anhänge). Es gibt zahlreiche Eigenschaften zu entdecken, aber wenig Neues zu lernen - man nutze die Spezifikationen einfach als Referenz.
 - Praxisnahe, kochbuchartige CSS/XML-Einführung in: „XML for the Word Wide Web“ von E. Castro.
 - CSS2 ist noch längst nicht hinreichend verbreitet
 - Es gibt Implementierungslücken und Unterschiede selbst in aktuellen Browsern.
 - Tip: Vergleichslisten suchen, nur Eigenschaften verwenden, die bereits hinreichende Unterstützung erfahren - etwa über Google mit Stichwörtern: „CSS compatibility“

CSS2: Anhang

Ergänzungen
Geplante Erweiterungen



- Vergleichstabelle: Welcher UA unterstützt was?
- Ergebnis einer kleinen Google-Suche:

<http://www.xs4all.nl/~ppk/css2tests/index.html>
- Diskussion der lokalen Kopie (Stand: 2003-04)
 - Ranking-Tabelle
 - Spezielle Features, insb. IE6 vs. Mozilla 6
- Noch zu vergleichen:
 - Analoge Tabelle für CSS1



- Neue Themen, geplant
 - „Blattaufteilung“:
 - Zugrunde liegendes Modell
 - Daraus folgende CSS-Parameter
 - Auswahl / Angebot mehrerer CSS-Dateien:
 - Wie verhält sich der UA?
 - Einbettung von CSS-Daten in XML (?) / XHTML
 - Ausblick zu CSS3