

LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
Department “Institut für Informatik”
Lehr- und Forschungseinheit Medieninformatik
Prof. Dr. Heinrich Hußmann

Projektarbeit

**Formatierung von wissenschaftlichen Publikationen mit CSS
und JavaScript**

Daniel Bertram
daniel.bertram@stud.ifi.lmu.de

Bearbeitungszeitraum: 1. 2. 2009 bis 31. 7. 2009
Betreuer: Dipl.-Medieninf. Raphael Wimmer
Verantw. Hochschullehrer: Prof. Dr. Heinrich Hußmann

Zusammenfassung

Diese Arbeit untersucht inwiefern gängige Formatierungen wissenschaftlicher Publikationen mit XHTML, CSS und JavaScript umgesetzt werden können. Nach einer kurzen Übersicht über konventionelle Formate und einer Beschreibung der verwendeten Technologien werden die gängigen Formatierungen wissenschaftlicher Publikationen näher betrachtet. Anschließend wird untersucht in wie weit diese Anforderungen erfüllen lassen. Dabei wird sowohl auf den sich in Entwicklung befindenden Standard CSS 3 eingegangen als auch auf die mit aktuellen Browsern tatsächlich vorhandenen Möglichkeiten. Abschließend folgt ein Fazit über den aktuellen Stand der Dinge sowie ein Ausblick auf zukünftige Möglichkeiten.

Abstract

This paper investigates how common formatting of scientific publications with XHTML, CSS and JavaScript can be implemented. After a brief overview of conventional formats and a description of the used technologies, common formatting of scientific publications are described in detail. Then it is examined how far these requirements can be met, considering both the upcoming standard CSS 3, as well as the existing possibilities with current browsers. Finally, a conclusion about the state of the art is given, combined with an outlook at future work.

Aufgabenstellung

Aktuelle wissenschaftliche Forschung wird oft als „Paper“ auf eine Konferenz eingereicht. Je nach Konferenz gibt es unterschiedliche Anforderungen an das Layout des Papers. Bisher werden die meisten Paper entweder mit LaTeX oder MS-Word gesetzt. Diese Projektarbeit soll untersuchen inwiefern gängige Formatierungen mittels HTML, CSS und JavaScript umgesetzt werden können. Je nach Ausgabemedium (Druck, Screen, Beamer, PDA), sollen unterschiedliche Darstellungen des Papers ausgegeben werden. Dabei muss die Druckausgabe den offiziellen Layoutvorgaben entsprechen. Wichtige Features sind mehrspaltiges Layout, Bildunterschriften und automatisches Literaturverzeichnis. Da die Paper oft überarbeitet werden, soll auch grundlegende Unterstützung für Überarbeitung implementiert werden.

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt, alle Zitate als solche kenntlich gemacht sowie alle benutzten Quellen und Hilfsmittel angegeben habe.

München, 21. Oktober 2009

.....

Inhaltsverzeichnis

1 Einleitung	1
2 Warum XHTML und CSS?	3
2.1 Konventionelle Formate	3
2.2 XHTML, CSS und JavaScript	3
2.3 Vorteile gegenüber konventionellen Formaten	4
2.4 Buchdruck mit PrinceXML	4
3 Gängige Formatierungen	7
3.1 Aufbau, mehrspaltiges Layout	7
3.2 Überschriften und Absätze	7
3.3 Bilder	9
3.4 Mathematische Formeln	9
3.5 Quelltext	9
3.6 Tabellen	9
3.7 Listen	9
3.8 Fußnoten	9
3.9 Papierformate und Seitenränder	10
3.10 Literaturverzeichnis	10
3.11 Ausgabe auf verschiedenen Medien	10
3.12 Zusammenfassung der Anforderungen	11
4 Formatierung mit XHTML und CSS	13
4.1 Textstrukturierung mit XHTML	13
4.2 XHTML Umsetzung	13
4.3 Verbindung von Inhalt und Darstellung	13
4.4 Elementanordnung	15
4.5 Textformatierung	18
4.6 Listenformatierung	18
4.7 Layouts für verschiedene Ausgabemedien	18
4.8 Automatisch erzeugte und veränderte Inhalte	19
4.9 Layout für seitenbasierende Medien	19
4.10 Mehrspaltige Layouts	20
4.11 Darstellung mathematischer Formeln	21
4.12 CSS Umsetzung	21
4.13 Eingeschränkte Browserunterstützung	21
5 JavaScript	25
5.1 Veränderungen des Dokumentbaumes	25
5.2 Mehrspaltiges Drucklayout	25
6 Fazit	29

1 Einleitung

Mit der fortschreitenden Entwicklung der zur Darstellung des World Wide Web verwendeten Technologien XHTML, CSS und JavaScript ist die Ausgabe nicht mehr auf konventionelle Monitore beschränkt. Alternative Ausgabegeräte wie Beamer oder mobile Geräte gewinnen zunehmend an Bedeutung. Zudem wird die Darstellung im Bereich der klassischen Ausgabegeräte, insbesondere im Druck, immer ausgereifter. Früher nur mit speziellen Textverarbeitungsprogrammen mögliche typografische Gestaltungsformen werden aufgrund weiterentwickelten Standards auf gängigen Browsern verfügbar.

Aktuelle wissenschaftliche Forschung wird oft als „Paper“ auf eine Konferenz eingereicht. Das Layout der „Papers“ ist vorgegeben, die Veröffentlichung der „Papers“ erfolgt konferenzbegleitend als Tagungsband („Proceedings“). Ziel dieser Arbeit ist es, am Beispiel der „Papers“ herauszufinden, wie weit diese Entwicklung tatsächlich fortgeschritten ist.

Dazu wird erstens untersucht inwiefern die, auf den Medientyp angepasste, Ausgabe auf verschiedenen Medien möglich ist. Zweitens wird am Beispiel Druck untersucht wie genau vorgegebene Formatierungen umgesetzt werden können. Dabei wird sowohl betrachtet was laut Standard möglich seien sollte, als auch was mit den tatsächlich verfügbaren Browsern im Moment möglich ist.

2 Warum XHTML und CSS?

Im folgenden Abschnitt werden Gründe dargestellt die für eine Formatierung wissenschaftlicher Publikationen mit XHTML, CSS und JavaScript sprechen. Dazu werden zuerst herkömmliche Formate kurz dargestellt und das Prinzip von XHTML, CSS und JavaScript erklärt. Anschließend werden die Vorteile der Formatierung wissenschaftlicher Publikationen mit XHTML, CSS und JavaScript im Vergleich zu herkömmlichen Formaten erläutert. Abschließend wird als Beispiel für die Fähigkeiten von CSS 3 im Druckbereich das Buch *Cascading Style Sheets: Designing for the Web* [24] beschrieben, welches nur mit XML und CSS erstellt wurde.

2.1 Konventionelle Formate

Als Grundlage dienen die Autorenrichtlinien für Konferenz- „Paper“ der Konferenzen „ACM Conference on Human Factors in Computing Systems“ (*SIGCHI Conference Proceedings, SIGCHI Extended Abstracts* [37]), „IEEE Information Visualization Conference“ (*InfoVis Research and Application Papers* [21]) und „Springer Lecture Notes in Computer Science“ (*LNCS Proceedings* [38]). Für alle vier Formate stehen sowohl *LaTeX*- als auch *MS Word*- Templates zur Verfügung, Abgabeformat ist das *Portable Document Format* (PDF)[2], ein durch die ISO (ISO 32000) standardisiertes Austauschformat.

Wissenschaftliche Publikationen werden als *LaTeX*- oder *MS Word*- Dateien gesetzt, aus diesen werden dann PDF-Dateien erzeugt. Das erzeugen von HTML-Dateien ist ebenfalls möglich, in *MS Word* kann direkt in diesem Format gespeichert werden, für *LaTeX* existieren dafür Konverter, beispielsweise Hevea [29]. Eine Konvertierung von PDF- oder HTML-Dateien in das Erstellungsformat ist allerdings nicht möglich.

Für das Öffnen von PDF-Dateien ist ein spezielles Programm nötig, beispielsweise der kostenlos erhältliche *Adobe Reader* der Firma Adobe.

2.2 XHTML, CSS und JavaScript

Die Extensible HyperText Markup Language [3] ist in ihrer aktuellen Form XHTML 1.1 (W3C Recommendation seit 31 Mai 2001) eine auf XML basierende Auszeichnungssprache. Der Dokumenteninhalt wird mit Hilfe von Elementen zu einem Dokumentbaum strukturiert und erhält durch sie eine gewisse semantische Auszeichnung.

XHTML ist modular aufgebaut [5] und kann bei Bedarf mit anderen Modulen oder XML basierenden Sprachen, wie zum Beispiel MathML [8], verknüpft werden. Je nach ausgewählten Modulen ergeben sich verschiedene XHTML Dokumenttypen [30]. Dadurch ist es möglich auch Elemente aus anderen Namensräumen, wie zum Beispiel mathematische Formeln, in XHTML Dokumenten zu verwenden.

Für die Darstellung des Dokumenteninhaltes sind Cascading Style Sheets [22] zuständig, für die einzelnen Elemente können umfangreiche Angaben zur Darstellung gemacht werden. Ein oder mehrere Stylesheets können mit einem XHTML Dokument verknüpft werden, falls kein Stylesheet angegeben ist wird das Standard Stylesheet des Webbrowsers verwendet.

CSS 2 [9] ist seit 1998 eine W3C Recommendation, die aktuelle Version 2.1 wurde zuletzt am 23.04.2009 aktualisiert. CSS 3 [7] ist modular aufgebaut und bietet wesentlich mehr Möglichkeiten als sein Vorgänger, insbesondere im Druckbereich. Einige Module befinden sich seit 1999 in der Entwicklung, der Status der Module ist ziemlich verschieden. Der genaue Status und Start der Entwicklung der einzelnen Module ist in Tabelle 2.1 dargestellt.

Die im Web Browser ausgeführte Skriptsprache JavaScript [14] (ECMA-262, 3. Edition Dezember 1999) ermöglicht der Zugriff auf die *Document Object Model* genannte Schnittstelle von XHTML-Dokumenten. Mit ihr ist es möglich Inhalt, Struktur und Darstellung von XHTML-Dokumenten dynamisch zu ändern.

Modulname:	in Entwicklung seit:	Status:
Media Queries	4 April 2001	Candidate Recommendation
Selectors	10 April 2000	Last Call
Generated and Replaced Content	14 Mai 2003	Working Draft
Multi-column Layout	22 Juni 1999	Working Draft
Paged Media	23 Juni 1999	Working Draft
Basic Box Model	26 Juli 2001	Working Draft
Fonts	31 Juli 2001	Working Draft
Generated Content for Paged Media	12 Juni 2006	Working Draft
Text	17 Mai 2001	Working Draft
Lists	20 Februar 2002	Working Draft
A MathML for CSS profile	27 April 2007	Working Draft

Tabelle 2.1: CSS 3 Module Status Juli 2009. [7]

Für die Anzeige von Webseiten existieren mehrere kostenlos erhältliche Webbrower, für diese Arbeit wurden die aktuellsten Versionen („Nightly-Builds“) von *Mozilla Firefox 3.5* [32], *Apple Safari 4* [4], [39], *Google Chrome 3* [16] und *Opera 10* [33] betrachtet.

2.3 Vorteile gegenüber konventionellen Formaten

Wissenschaftliche Publikationen werden herkömmlicherweise als PDF-Dateien auf Webseiten veröffentlicht und können dort mit Hilfe eines speziellen Programmes, beispielsweise dem Adobe Reader, geöffnet werden. Alternativ können auch aus *LaTeX*- oder *MS Word*-Dateien HTML-Dateien erzeugt werden, deren Layout allerdings nur eingeschränkt angepasst werden kann.

Im Gegensatz dazu können mit XHTML und CSS erstellte und veröffentlichten „Papers“ direkt im Webbrower betrachtet werden, es erfolgt kein Medienbruch. Durch die Trennung von Inhalt und Darstellung ist die medienspezifisch angepasste Ausgabe auf verschiedenen Medien möglich [28], ohne dass dazu der Inhalt kopiert werden muss. Barrierefreie Veröffentlichung [10] ist ebenfalls möglich, der Nutzer kann Beispielsweise durch Nutzer-CSS die Darstellung an seine Bedürfnisse anpassen. Des weiteren existieren für den verbreiteten Standard XHTML auch spezielle Ausgabegeräte, wie zum Beispiel Screenreader. Dadurch dass Erstellungsformat und Veröffentlichungsformat identisch sind lassen sich mit XHTML und CSS veröffentlichte wissenschaftliche Publikationen leichter bearbeiten, da jederzeit auf das Erstellungsformat zugegriffen werden kann.

Inhalte können semantisch Ausgezeichnet werden, beispielsweise mit Hilfe von *Microformats* [31] oder *RDFa*-Tags [1], dadurch ist eine einfache maschinelle Weiterverarbeitung der Daten, insbesondere durch Suchmaschinen, möglich. Auch die Ausgabe als PDF-Datei ist möglich, wodurch die Rückwärtskompatibilität bezüglich des Abgabeformates gegeben ist.

2.4 Buchdruck mit PrinceXML

Um die Fähigkeiten von CSS 3 im Druckbereich zu verdeutlichen ist das Beispiel des Buches *Cascading Style Sheets: Designing for the Web*, welches mit PrinceXML [40] erstellt wurde, gut geeignet.

PrinceXML ist ein XML-Konverter, der sowohl XHTML als auch CSS 3 zum Großteil unterstützt und daraus PDF-Dateien erzeugt. Hakon Wium Lie und Bert Bos haben das Buch *Cascading Style Sheets: Designing for the Web* mit XHTML und CSS 3 erstellt und die zum Druck benötigte PDF-Datei mit PrinceXML erzeugt [25].

Neben den CSS 3 Eigenschaften, welche in Abschnitt 4 näher beschrieben werden, machten sie ausgiebig vom XHTML class-attribut Gebrauch. Mit dessen Hilfe erstellten sie ein eigenes Microformat für ihr Buch, *boom!* genannt. Ein Microformat ist ein Format zur semantischen Aus-

zeichnung von XHTML [31], damit konnten sie nicht in XHTML vorhandene aber für die Struktur eines Buches benötigte Elemente auszeichnen.

Das Beispiel zeigt, dass es möglich ist XHTML mit CSS 3 als Dokumentformat auch für den Druckbereich zu nutzen, zumindest mit Hilfe des Konverters PrinceXML. Es dient als Bestätigung für die Aufgabenstellung, wenn theoretisch die Erstellung eines ganzen Buches nur mit XHTML und CSS möglich ist, sollte auch die (nicht so umfangreiche) Formatierung von wissenschaftlichen Publikationen mit XHTML und CSS für verfügbaren Webbrowser möglich sein.

3 Gängige Formatierungen

Im folgenden werden gängige Formatierungen wissenschaftlicher Publikationen dargestellt und sich darauf ergebende Anforderungen abgeleitet. Als Grundlage dienen die in 2.1 erwähnten Autorenrichtlinien für Konferenz- „Paper“.

3.1 Aufbau, mehrspaltiges Layout

Der Aufbau der untersuchten „Paper“ ist relativ ähnlich. Auf die Überschrift und Informationen zu den Autoren folgen die Zusammenfassung sowie Schlüsselwörter. Diese Bereiche sind bei den Formaten *LNCS Proceedings* und *InfoVis Research and Application Papers* im Vergleich zum folgenden Inhalt eingerückt.

Während bei *LNCS Proceedings* der Inhalt einspaltig ist und direkt folgt, kann bei *InfoVis Research and Application Papers* nach den Autoren optional ein sogenanntes „Teaser Image“ folgen. Auf die Schlüsselwörter folgt eine „Diamond Rule“ genannte Trennlinie, anschließend beginnt der zweispaltige Inhaltsteil.

Bei den Formaten *SIGCHI Conference Proceedings* und *SIGCHI Extended Abstracts* ist nur die Überschrift vorgestellt. Autoreninformationen, Zusammenfassung und Schlüsselwörter sind Bestandteil des zweispaltigen Layouts, wobei den Autoreninformationen bei *SIGCHI Extended Abstracts* die gesamte erste Spalte gehört und sie bei *SIGCHI Conference Proceedings* in einem eigenständigen zweispaltigen Absatz untergebracht sind.

Die Formate *SIGCHI Conference Proceedings*, *SIGCHI Extended Abstracts* und *InfoVis Research and Application Papers* besitzen außerdem noch einen abgesetzten Absatz fester Größe am unteren Rand der ersten Spalte, welcher Informationen zum Urheberrecht enthält. Abbildung 3.1 zeigt die erste Seite der vorgestellten Formate.

Als Anforderungen ergibt sich hieraus, dass ein mehrspaltiges Layout mit von der Größe der Seite und der vorgestellten Elemente abhängigen Spaltenhöhen nötig ist. Spaltenumbruch bei Seitenende sowie spaltenübergreifender Textfluss mit Textumbruch sind ebenso erforderlich. Beim Textumbruch ist es erforderlich, dass weder *Schusterjungen* (Seitenumbruch nach der ersten Zeile eines Absatzes) noch *Hurenkindern* (Seitenumbruch vor der letzten Zeile eines Absatzes) entstehen.

3.2 Überschriften und Absätze

Alle untersuchten „Paper“ verwenden Überschriften verschiedener Gliederungstiefen sowie Absätze. Schriftart, Größe, Formatierung, Zeilenabstand und der Abstand zwischen den Absätzen und Überschriften ist bei den untersuchten „Papers“ unterschiedlich.

Überschriften sind den Absätzen vorgestellt, bei *LNCS Proceedings* teilweise auch als erstes Wort in den Textfluss des Absatzes integriert. Sie sind linksbündig ausgerichtet und bei *InfoVis Research and Application Papers* und *LNCS Proceedings* auch durchgehend nummeriert.

Absätze sind im Blocksatz, nur bei *SIGCHI Extended Abstracts* ist der Text linksbündig ausgerichtet. Bei *InfoVis Research and Application Papers* und *LNCS Proceedings* sind die Absätze, bis auf den direkt auf eine Überschrift folgenden Absatz, eingerückt.

Textformatierung beschränkt sich auf die gängigen Auszeichnungsarten Fett, Kursiv, dickten-gleiche Schrift und Kapitälchen.

Daraus ergibt sich, dass verschiedene Schriftarten, Größen, Ausrichtungen, Formatierungen und Abstände individuell für jedes Element möglich sein müssen, ebenso wie die Nummerierung der Überschriften.

3.2 Überschriften und Absätze

3 GÄNGIGE FORMATIERUNGEN

Global Illumination for Fun and Profit

Ray G. Biv, Ed Grimley, Member, IEEE, and Martha Stewart



Fig. 1. In the Clouds: Vancouver from Cypress Mountain

Abstract—*Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, sed diam nonummy nibus etiam tempor inibunt ut labore et dolore magna aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, utero et eros et acussum et isto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore feugiat nulla facilis.*

Index Terms—Radioactivity, global illumination, constant time

1 INTRODUCTION

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibus etiam tempor inibunt ut labore et dolore magna aliquip ex ea commodo consequat, utero et eros et acussum et isto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore feugiat nulla facilis. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl aliqip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, utero et eros et acussum et isto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore feugiat nulla facilis.

2 EXPOSITION

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibus etiam tempor inibunt ut labore et dolore magna aliquip ex ea commodo consequat, utero et eros et acussum et isto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore feugiat nulla facilis.

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse

molestie consequat, utero et eros et acussum et isto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore feugiat nulla facilis [1].

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibus etiam tempor inibunt ut labore et dolore magna aliquip ex ea commodo consequat, utero et eros et acussum et isto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore feugiat nulla facilis.

• Roy G. Biv is with Starbucks Research. E-mail: rog.biv@starbucks.com

• Ed Grimley is with Grinder Widgets, Inc. E-mail: ed.grimley@grinderwidgets.com

• Martha Stewart is with Martha Stewart Enterprises at Microsoft Research. E-mail: Martha.Stewart@microsoft.com

Manuscript received 31 March 2009; revised 27 July 2009; posted online 11 October 2009; mailed on 5 October 2009.

For information on obtaining reprints of this article, please send email to: tvcg@computer.org

(a)

Lecture Notes in Computer Science: Authors' Instructions for the Preparation of Camera-Ready Contributions to LNCS/LNAI/LNBI Proceedings

Alfred Hofmann*, Brigitte Apfel, Ursula Barth, Christine Günther,
Ingrid Haas, Frank Holzwarth, Anna Kramer, Leonie Kunz,
Nikola Sator, Erika Siebert-Cole, and Peter Sträßer
Springer-Verlag, Computer Science Editorial,
Türgartenstr. 17, 69119 Heidelberg, Germany
(*alred.hofmann@springer.de, brigitte.apfel@springer.de, christine.guenther.
ingrid.haas, frank.holzwarth, anna.kramer, leonie.kunz, nikola.sator,
erika.siebert-cole, peter.straeßer, lnccs@springer.com
<http://www.springer.com/lncs>

Abstract. The abstract should summarize the contents of the paper and should contain at least 70 and at most 150 words. It should be written using the *abstract* environment.

Key words: We would like to encourage you to list your keywords within the abstract section

1 Introduction

You are strongly encouraged to use *BTEX* 2r, for the preparation of your camera-ready manuscript together with the corresponding Springer class file *lncs.cls*. Only if you use *BTEX* 2r can hyperlinks be generated in the online version of your manuscript.

The *BTEX* 2r class file for LNCS is located in the “*lncs*” subdirectory in *tex/texmf-local/tex/dei/publish/tex/lncs/ltx2e/* and entitled *lncs2e.instruc* and entitled *lncs2e.tex*. There is a separate package for Word users. Kindly send the final and checked source and PDF files of your paper to the Contact Volume Editor. This is usually one of the organizers of the conference. You should make sure that the *BTEX* and the PDF files are original and correct and that only one version of your paper is sent. It is not possible to update files at a later stage. Please note that we do not need the printed paper.

* Please note that the LNCS Editorial assumes that all authors used the western naming convention, with given names preceding surnames. This determines the structure of the names in the running heads and the author index.

(c)

SIGCHI Conference Proceedings Format

First Author Name (Blank if Blind Review)

Affiliation (Blank if Blind Review)

Address (Blank if Blind Review)

e-mail address (Blank if Blind Review)

Optional phone number (Blank if Blind Review)

Second Author Name (Blank if Blind Review)

Affiliation (Blank if Blind Review)

Address (Blank if Blind Review)

e-mail address (Blank if Blind Review)

Optional phone number (Blank if Blind Review)

ABSTRACT

In this paper we describe the formatting requirements for SIGCHI Conference Proceedings, and some recommendations on writing for the worldwide SIGCHI readership. Please review this document even if you have submitted to SIGCHI conferences before, for some format details have changed relative to previous years. These include the formatting of table captions, the formatting of references, and a requirement to include ACM DL indexing information.

Author Keywords

Guides, instructions, author's kit, conference publications.

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI); Miscellaneous.

INTRODUCTION

This format is to be used for submissions that are published in the conference proceedings. We wish to give this volume a consistent, high-quality appearance. We therefore ask that authors follow some simple guidelines. In essence, you should format your paper exactly as you would normally do for publication from the conference web site, and replace the content with your own material. The template file contains specially formatted styles (e.g., Normal, Heading, Bullet, Table Text, References, Title, Author, Affiliation) that will reduce your work in formulating your submission.

PAGE SIZE AND COLUMNS

On each page there must be a margin of 18 x 23.5 mm (7.1 x 9.25 in.) centered on a U.S. letter page, beginning 1.9 cm (.75 in.) from the top of the page with a .85 cm (.33 in.)

space between two 8.4 cm (3.3 in.) columns. On an A4 page, use a text area of the same dimensions (18 x 23.5 cm), again centered. Right margins should be justified, not ragged. Beware, especially when using this template on a Macintosh. Word can change these dimensions in unexpected ways.

TYPESET TEXT

Preferably, use a typesetting system or a word processor or typesetter. Please note that page layout may change slightly depending upon the printer you have specified. For this document, printing to Adobe Acrobat PDF Writer was specified. In the resulting page layout, Figure 1 appears at the top of the left column on page 2, and Table 1 appears at the top of the right column on page 2. You may need to reposition the figures if your page layout or PDF-generation software is different.

Title and Authors

Your paper's title, authors and affiliations should run across the full width of the page in a single column (7.8 cm (7 in.) wide). The title should be in Hebrew 18-point bold; use Arial if Helvetica is not available. Authors' names should be in Times Roman 12-point bold, and affiliations in Times Roman 12-point (note that Author and Affiliation are defined styles in this template file).

To position names and addresses, use a single tab stop, and only one address is needed, use a centered tab stop to center all name and address text on the page; for two addresses, use two centered tab stops, and so on. For more than three authors, you may have to place some address information in a footnote, or in a named section at the end of your paper. Please use the standard abbreviations and telephone dialing prefixes. Leave one 10-pt line of white space below the last line of the page.

Abstract and Keywords

Every submission should begin with an abstract of about 150 words, followed by a set of keywords. The abstract and keywords should be placed in the left column of the first page under the left half of the title. The abstract should be a concise statement of the problem, approach and conclusions of the work described. It should clearly state the paper's contribution to the field of HCI.

(b)

CHI 2009 Extended Abstracts Template

Abstract
In this paper we describe the formatting requirements for CHI 2009 Extended Abstracts and offer recommendations on writing for the worldwide SIGCHI readership.

Keywords
publications, instructions, author's kit, conference publications.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI); Classification system.

Introduction
This template is to be used for submissions that are intended for the conference extended abstracts. We wish to give this column a consistent, high-quality appearance. We therefore ask that authors follow some simple guidelines. In essence, you should format your paper exactly as you would normally do for publication from the conference web site, and replace the content with your own material. The easiest way to do this is simply to download a template from the conference website and replace the content with your own material. Copyright is held by the author(s) or publisher, who is identified in the page header. ACM is identifying the copyright holder in this page header only for the convenience of the author(s) or publisher. ACM's Terms of Use (http://www.acm.org/publications/rights/terms.html) apply.

Table 1
Copyright is held by the author(s) or publisher(s).
ACM 978-1-60538-246-7/09/04. \$5.00.

(d)

Abbildung 3.1: Paper Templates: *InfoVis Research and Application Papers* (a) [21], *SIGCHI Conference Proceedings* (b) [37], *LNCS Proceedings* (c) [38], *SIGCHI Extended Abstracts* (d) [37].

3.3 Bilder

Bei allen untersuchten Formaten ist das Einfügen von Bildern möglich. Bei *LNCS Proceedings* und *SIGCHI Extended Abstracts* werden diese im Textfluss dargestellt, bei *InfoVis Research and Application Papers* und *SIGCHI Conference Proceedings* müssen sie am Anfang oder Ende einer Spalte angebracht werden.

Bei *InfoVis Research and Application Papers* darf der Zusammenfassung ein „Teaser Image“ vorgestellt werden, welches nicht wie die anderen Bilder angeordnet wird. Bei *SIGCHI Extended Abstracts* ist außerdem das Einfügen eines ganzseitigen Bildes möglich, ebenso das Einfügen eines Bildes in den linken Seitenrand. Beim Spalten- und Textumbruch wird diese Bilderseite übersprungen, der Textfluss geht auf der nächsten Seite weiter.

Bei allen Formaten besitzen die Bilder Unterschriften welche durchgängig nummeriert und zusammen mit den Bildern angeordnet werden müssen. Daher ist eine Bildanordnung und Nummerierung nötig, die je nach Format, Position oder Typ die Bilder entsprechend den Vorgaben anordnet.

3.4 Mathematische Formeln

Mathematische Formeln sind ebenfalls Bestandteil wissenschaftlicher Publikationen. Wie bei Bildern auch müssen sie je nach Format im Textfluss oder am Anfang beziehungsweise Ende einer Spalte angeordnet werden. Neben der Anordnung ist auch eine Nummerierung nötig.

3.5 Quelltext

Vorgaben zur Darstellung von Quelltext werden nur bei dem Format *LNCS Proceedings* gemacht. Dort werden Absätze mit Quelltext wie normaler Text angeordnet und umgebrochen. Der Quelltext muss linksbündig ausgerichtet sein und eine dicktengleiche Schriftart verwenden. Textzeilen müssen nach Bedarf eingerückt werden können.

3.6 Tabellen

Auch Tabellen sind ein fester Bestandteil wissenschaftlicher Publikationen. Genau wie Bilder müssen sie bei *LNCS Proceedings* und *SIGCHI Extended Abstracts* im Textfluss dargestellt werden, bei *InfoVis Research and Application Papers* und *SIGCHI Conference Proceedings* müssen sie am Anfang oder Ende einer Spalte angebracht werden.

Tabellen besitzen Überschriften welche durchgängig nummeriert und automatisch zusammen mit den Tabellen angeordnet werden müssen. Bei *InfoVis Research and Application Papers* und *LNCS Proceedings* werden sie den Tabellen vorgestellt, bei den anderen Formaten werden sie wie Bildunterschriften angeordnet.

Dicke und Darstellung der Ränder sowie Innenabstände der Tabellen sind auch, je nach Format, unterschiedlich. Daher ist eine formatspezifische Tabellenformatierung, Anordnung und Nummerierung nötig.

3.7 Listen

Listen werden bei allen Formaten wie normaler Text im Textfluss angeordnet und umgebrochen. Es kommen neben Nummerierungen auch verschiedene Aufzählungszeichen bei der Formatierung zum Einsatz. Deshalb müssen Listenformatierungen individuell vorgegeben werden können.

3.8 Fußnoten

Fußnoten sind in jedem der Formate möglich. Sie werden am Ende der betreffenden Spalte angebracht und mit hochgestellten Zahlen durchnummeriert. Ihnen vorgestellt ist eine Trennlinie

die je nach Format in Länge, Dicke, Abstand und Anordnung variiert. Anordnung, Einfügen der Trennlinie und Nummerierung der Fußnoten muss je nach „Paper“-Typ unterschiedlich erfolgen können.

3.9 Papierformate und Seitenränder

Jede der Formatvorlagen besitzt einen Inhaltbereich mit fester Größe, ein vorgegebenes Papierformat sowie feste Seitenränder. Diese sind entweder direkt angegeben oder lassen sich aus den anderen Größen errechnen. Die Ausrichtung des „Paper“-Types *SIGCHI Extended Abstracts* ist Querformat, bei den anderen drei Typen Längsformat.

Größenangaben sind in Zentimetern und Inch, wobei teilweise nur ein Wert der passende Ausgangswert ist und der andere nur eine ungenau gerundete Umrechnung. Bei dem Format *SIGCHI Conference Proceedings* beispielsweise sind alle Zentimeter angaben gerundet, was zu erheblichen Abweichungen führt. Das Papierformat ist bei *LNCS Proceedings* „Din A4“ (210 mm x 297 mm), bei den anderen Formaten das amerikanische „Letter“ Format (216 mm x 279 mm). Insbesondere der Breitenunterschied kann in Verbindung mit festen Seitenrändern beim Ausdruck im jeweils anderen Format zu unschönen Skalierungen führen. Daher ist neben der Festlegung der Ausrichtung und der Einstellung der verschiedenen Bereichsgrößen eine an das jeweilige Format angepasste Randgröße erforderlich.

3.10 Literaturverzeichnis

Als Format für Literaturverzeichnisse wird bei allen „Paper“-Typen *BibTeX* [34] verwendet. *BibTeX* ist ein von Oren Patashnik entwickeltes Programm zur Erstellung von Literaturverzeichnissen in *LaTeX*. Es verknüpft im Text vorkommende Zitatverweise mit einer Literaturdatei und erstellt daraus ein Literaturverzeichnis. Einen Ausschnitt aus einer Literaturdatei sehen Sie in Abbildung 3.2.

```

@misc{Ado2009,
  title = {Adobe Acrobat Reader 7},
  url = {http://www.adobe.com/products/acrobat/},
}

@article{And1992,
  author = {Anderson, R.E.},
  title = {Social impacts of computing: Codes of professional ethics},
  journal = {Social Science Computing Review 10},
  number = {2 (1992)},
  pages = {453-469},
}

```

Abbildung 3.2: Ausschnitt einer Literaturdatei

Daher ist für die Formatierung wissenschaftlicher Publikationen ein *BibTeX* Parser erforderlich der Literaturdateien einlesen kann. Diese Einträge müssen dann formatiert und entsprechend ihrer Verwendung im Text sortiert werden.

3.11 Ausgabe auf verschiedenen Medien

Da wissenschaftliche Publikationen primär als „Paper“ veröffentlicht werden, beziehen sich die in diesem Abschnitt betrachteten Formatvorlagen nur auf das Medium Papier. Für die anderen Medien existieren keine Vorgaben, spezifische Formatierungen für andere Medien sind auch nicht Teil dieser Arbeit. Lediglich die Möglichkeit medienspezifischer Formatierungsangaben ist erforderlich.

3.12 Zusammenfassung der Anforderungen

Zusammenfassend ergeben sich für die Formatierung wissenschaftlicher Publikationen folgende Anforderungen, welche mit Hilfe von XHTML, CSS und JavaScript erfüllt werden müssen:

1. Unterstützung verschiedene Formatierungen auch für Elemente des gleichen Typs.
2. Aufteilung der Elemente in ein mehrspaltiges Layout mit spaltenübergreifenden Textfluss.
3. automatische Nummerierung von Elementen.
4. automatische Anordnung von Elementen.
5. Darstellung mathematischer Formeln.
6. verschiedene Papierformate und Ausrichtungen.
7. Verarbeitung von Literaturdateien, Erstellung eines Literaturverzeichnisses.
8. verschiedene Formatierungen für verschiedene Medien.

4 Formatierung mit XHTML und CSS

Im folgenden werden die für die Formatierung von wissenschaftlichen Publikationen wichtigen Bestandteile von XHTML und CSS dargestellt.

Zuerst werden die für die in Abschnitt 3 ermittelten Anforderungen relevanten XHTML-Elemente kurz erläutert, anschließend werden die vielfältigen Möglichkeiten der Darstellung mit CSS betrachtet.

Abschließend folgt eine Zusammenfassung in der auf die Möglichkeiten und Limitierungen, insbesondere in zur Zeit verfügbaren Webbrowsern, eingegangen wird.

4.1 Textstrukturierung mit XHTML

XHTML [35] bietet zur Auszeichnung von Dokumenten verschiedene Elemente, die für wissenschaftliche Publikationen wichtigsten werden im folgenden kurz dargestellt.

Die meisten Elemente umschließen den Inhalt in der Form `<tag>Inhalt</tag>`, alleinstehende Elemente werden in der Form `<tag/>` geschrieben.

Zur Auszeichnung von Überschriften verschiedener Ordnung existieren die Elemente `<h1>` bis `<h6>`, Absätze werden mit `<p>` ausgezeichnet.

Bei Listen wird zwischen `` für geordnete und `` für ungeordnete Listen unterschieden. Die einzelnen Listeneinträge werden mit `` abgegrenzt.

Tabellen werden mit `<table>` ausgezeichnet, sie können desweiteren in einen Kopf (`<thead>`), Rumpf (`<tbody>`) und Fußbereich (`<tfoot>`) unterteilt werden. Einzelne Tabellenzeilen werden mit `<tr>` ausgezeichnet, diese enthalten die Tabellenzellen. Kopfzellen können mit `<th>` ausgezeichnet werden, normale Datenzellen mit `<td>`. Für Tabellenüberschriften gibt es das `<caption>` Element.

Quelltext wird mit dem `<code>` Element ausgezeichnet, Zitate mit `<blockquote>` und Literaturverweise mit `<cite>`.

`<div>` Elemente dienen der Gruppierung beliebiger Elemente, `` Elemente zum Einbinden von Bildern und `<object>` Elemente zum Einbinden von Objekten verschiedener Art.

Für die Überarbeitung von XHTML Dokumenten gibt es das Element `` zur Markierung gelöschten Textes und `<ins>` zur Auszeichnung von nachträglich eingefügtem Text.

Des Weiteren gibt es noch Elemente zur Formatierung, welche Funktionen herkömmlicher Textverarbeitungsprogramme bereitstellen. Dazu zählen unter anderem kursiv (`<i>`), fett (``), diktengleich (`<tt>`), hochgestellt (`<sup>`) und tiefgestellt (`<sub>`).

Zeilenumbrüche können mit `
` erzwungen werden, Trennlinien werden mit `<hr>` erzeugt.

Bei Bedarf kann einem Element auch eine Klasse oder Identität zugewiesen werden.

4.2 XHTML Umsetzung

Neben den im vorherigen Abschnitt (4.1) beschriebenen XHTML Elementen werden zur Formatierung wissenschaftliche Publikationen weitere Elemente benötigt.

Daher werden mit Hilfe des class-attributes eigene Elemente für wissenschaftliche Publikationen erstellt, unter anderem für Informationen zu den Autoren, die Zusammenfassung und Schlüsselwörter. Die Elemente sind in Tabelle 4.1 aufgeführt.

Einen Ausschnitt der XHTML Seite des Templates des Formates *InfoVis Research and Application Papers* sehen Sie in Abbildung 4.1

4.3 Verbindung von Inhalt und Darstellung

XHTML-Elemente werden durch CSS mit Darstellungseigenschaften verknüpft, die Auswahl der Elemente erfolgt über Selektoren [11].

Klassenname:	Verfügbar für:	Erklärung:
abstract	Absätze und Überschriften	Zusammenfassung
keywords	Absätze und Überschriften	Schlüsselwörter
acmkeywords	Absätze und Überschriften	ACM Schlüsselwörter
author	Absätze	Informationen zu den Autoren
figCaption	<div> Elemente	Bildunterschrift
mathCaption	<div> Elemente	Formelunterschrift
footnote	<div> Elemente	Fußnote
copyrightbox	<div> Elemente	Copyright Box

Tabelle 4.1: Elemente für wissenschaftliche Publikationen.

```

<body>
  <h1>Global Illumination for Fun and Profit</h1>
  <p class = "author">
    Roy G. Biv, Ed Grimley, <i>Member, IEEE,</i> and Martha Stewart
  </p>
  <h2 class = "abstract">Abstract</h2>
  <p class = "abstract">
    –Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse
    molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero
    eros et accumsan et iusto odio dignissim qui blandit praesent luptatum
    zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum
    dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh
    euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.<br/>
    And this is what references look like <cite>ware:2004:IVP</cite><br/>
    Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper
    suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis
    autem vel eum iriure dolor in hendrerit in vulputate velit esse
    molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero
    eros et accumsan et iusto odio dignissim qui blandit praesent luptatum
    zzril delenit augue duis dolore te feugait nulla facilisi.
  </p>
  <h2 class = "keywords">Index Terms</h2>
  <p class = "keywords">
    –Radiosity, global illumination, constant time.
  </p>
  <h2>Introduction</h2>
  <p>
    Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
    eirmod tempor invidunt ut labore et dolore magna aliquyam erat,
    sed diam voluptua. At vero eos et accusam et justo duo dolores et ea
    rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem
    ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing
    elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore
    magna aliquyam erat, sed diam voluptua <cite>kitware2003</cite><br/>
    At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren,
    no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit
    amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt
    ut labore et dolore magna aliquyam erat, sed diam voluptua. At
    vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd
    gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.
  </p>

```

Abbildung 4.1: Ausschnitt einer XHTML Seite des Formates *InfoVis Research and Application Papers*.

Selektoren sind Muster die mit Elementen des Dokumentbaumes abgeglichen werden. Selektoren werden in einfache Selektoren und Kombinatoren unterteilt.

Einfache Selektoren sind neben dem Universalselektor (*), Typ- (zum Beispiel *p*) und Attributselektoren (zum Beispiel *h1[title]*).

Kombinatoren werden ihrerseits wieder in Nachfahren-, Kinder- und Geschwisterkombinatoren unterteilt. Nachfahrenkombinator ist das Leerzeichen, Beispielsweise passt *h1 em* auf alle ** Elemente die Nachfahren von *<h1>* Elementen sind. Kinderkombinator ist *>*, Geschwisterkombinatoren sind *~* für Geschwister und *+* für direkte Geschwister.

Aus einfachen Selektoren können zusammen mit Kombinatoren Selektorenketten gebildet werden, wodurch sich vielfältige Auswahlmöglichkeiten ergeben.

Beispielsweise können mit *h2 + p {...}* alle *<p>* Elemente, die direkt auf eine Überschrift zweiter Ordnung folgen, ausgewählt und mit Darstellungseigenschaften versehen werden.

Selektoren mit gleicher Deklaration können gruppiert werden. So ergibt zum Beispiel *h1,h2{font-family: „Times New Roman“}* das gleiche wie *h1 {font-family: „Times New Roman“}* *h2 {font-family: „Times New Roman“}*.

Eine weitere Art von Selektoren in CSS sind Attributselektoren. Diese gleichen nicht nur Elemente sondern auch ihre Attribute ab. Zum Beispiel passt *div[class=“test”]* auf alle *<div>* Elemente aus deren class-attribut den Wert „test“ hat.

Für das oft verwendete class-attribut gibt es eine Kurzschreibweise bestehend aus einem Punkt und dem Klassennamen. Daher lässt sich das vorherige Beispiel auch als *div.test* schreiben.

CSS bietet auch die Möglichkeit Elemente nach ihrer Identität auszuwählen. Diese Selektoren werden mit dem Rautenzeichen gekennzeichnet. Beispielsweise wählt *div#test* das *<div>* Element aus dessen Attribut *id=“test“* ist.

Die letzte Art von Selektoren sind die sogenannten Pseudoklassen-Selektoren. Sie ermöglichen die Auswahl anhand von Informationen die über die Elemente des Dokumentbaumes hinausgeht.

Pseudoklassen werden als Doppelpunkt gefolgt vom Namen geschrieben. So kann beispielsweise mit *p:first-of-type{...}* das *<p>* Element ausgewählt werden welches das erste Kind seines Elternelementes vom Typ *<p>* ist.

CSS bietet auch die Möglichkeit auf Teile von Elementen zuzugreifen für die in XHTML keine Auszeichnung existiert, sowie auf automatisch erstellte Inhalte die nicht im Dokument enthalten sind. Dadurch ist eine sehr spezifische Auswahl möglich, beispielsweise kann mit *p::first-letter* der erste Buchstabe eines Absatzes ausgewählt und besonders dargestellt werden.

Ein Sonderfall sind die *::before* und *::after* Pseudoelemente, womit automatisch erzeugte Inhalte formatiert werden können. Auf automatisch erzeugte Inhalte wird in 4.8 noch näher eingegangen.

Darstellungseigenschaften in CSS folgen auf Selektoren. Sie werden in geschweifte Klammern eingeschlossen, mit Strichpunkt getrennt und als Attribut-Wert Paare formuliert, wobei es auch einige Kurzschreibweisen gibt auf die hier nicht weiter eingegangen wird.

Ein einfaches Beispiel welches allen Textabsätzen die Schriftart „Helvetica“ und den Schriftstil „italic“ zuweist ist *p {font-family: „Helvetica“; font-style: „italic“}*.

4.4 Elementanordnung

CSS stellt Dokumente als hierarchisch ineinander geschachtelten Boxen [6] dar, siehe Abbildung 4.2.

Jede Box besteht aus den Bereichen Inhalt (*content*), Innenabstand (*padding*), Rahmen (*border*) und Außenabstand (*margin*), dargestellt in Abbildung 4.3.

Es wird zwischen verschiedenen Darstellungsarten von Boxen unterschieden, sie werden mit dem „display“ Attribut gesetzt.

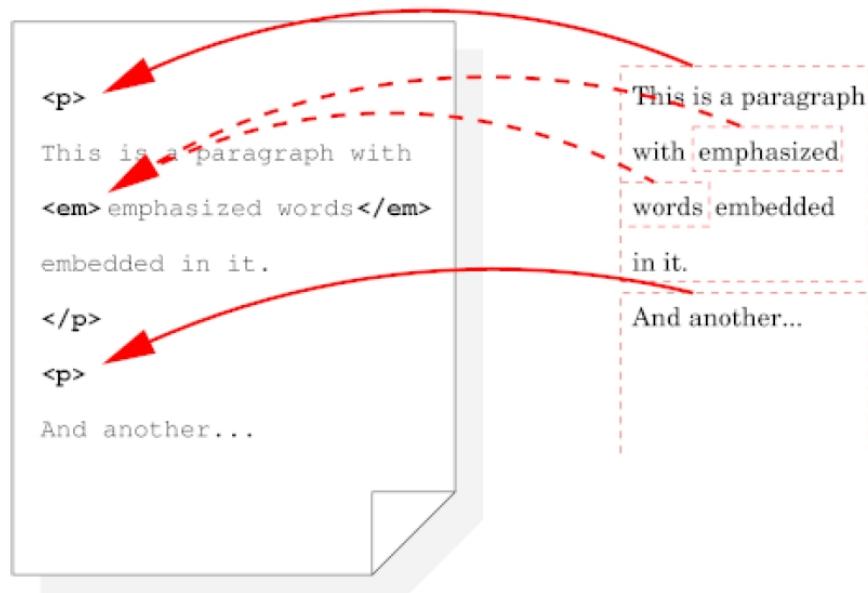


Abbildung 4.2: CSS Elementdarstellung durch Boxen. [6]

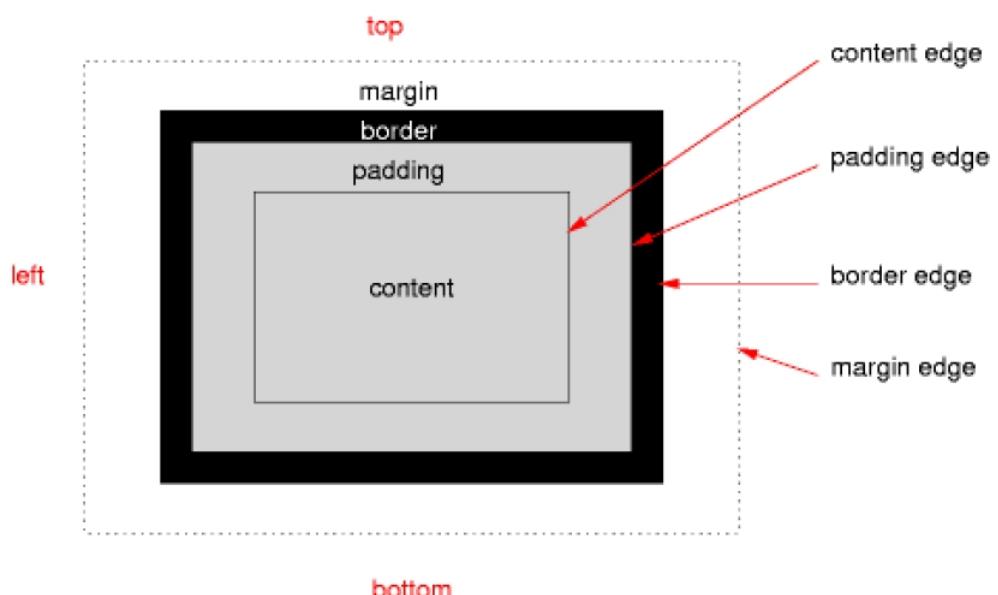


Abbildung 4.3: Die verschiedenen Bereiche einer CSS Box. [6]

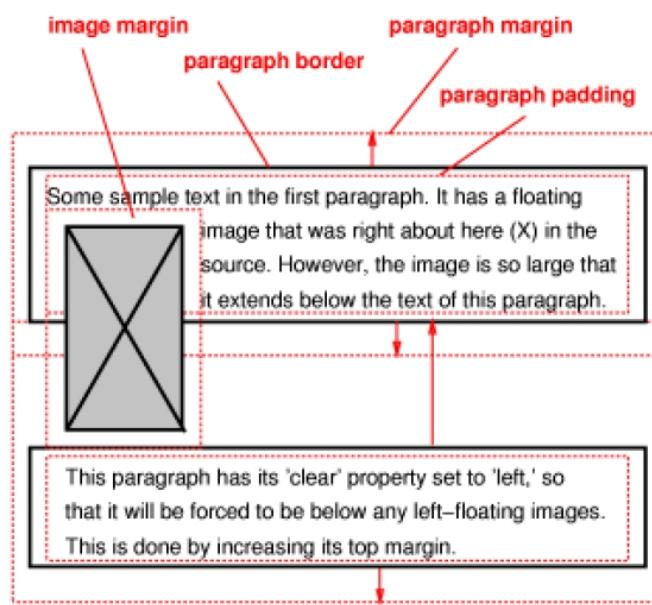


Abbildung 4.4: Darstellung der CSS Eigenschaften „float“ und „clear“. [6]

Die für die Formatierung von wissenschaftlichen Publikationen wichtigsten Darstellungsarten sind *display:inline*, *display:block* und *display:run-in*.

„Inline“ Elemente werden im normalen Textfluss der Elternbox dargestellt, „Block“ Elemente hingegen werden als eigenständig betrachtet und dementsprechend dargestellt. „Run-in“ Elemente sind ein Sonderfall, sie erlauben es zwei aufeinanderfolgende Blockelemente zu verschmelzen. Beispielsweise wird eine als „Run-in“ deklarierte Überschrift als erstes Wort des nachfolgenden Absatzes dargestellt.

Jedes XHTML-Element hat eine vorgegebene Darstellungsart, ** Elemente beispielsweise werden „Inline“ dargestellt, *<p>* Elemente als „Block“.

Für die Größe jeder Box können Höhe (*width*) und Breite (*height*), auch als Mindest- (*min-width*, *min-height*) und Höchstwert (*max-width*, *max-height*), angegeben werden. Größenangaben können absolut oder als Prozentsatz der verfügbaren Fläche angegeben werden.

Die Größe von Innenabstand (*padding-top*, *padding-right*, *padding-bottom*, *padding-left*) Rahmen (*border-top-width*, *border-right-width*, *border-bottom-width*, *border-left-width*) und Außenabstand (*margin-top*, *margin-right*, *margin-bottom*, *margin-left*) können ebenfalls für jede Seite einer Box angegeben werden.

Innenabstände können für jedes Element angegeben werden, ebenso Rahmendicke, Art (*border-style*) und Farbe (*border-color*).

Außenabstände können für die meisten Elemente angegeben werden, mit Ausnahme von Tabellen und einigen „Inline“ Elementen. Außenabstände benachbarter Boxen können verschmolzen werden (*margin-collapse*).

Ein weiteres wichtiges Werkzeug zu Anordnung von Elementen ist die Möglichkeit der Manipulation des Elementflusses. Mit dem Attribut *float* kann angegeben werden ob die Box links (*left*) oder rechts (*right*) angeordnet werden soll, Inhalt fließt dann auf der freien Seite an der Box vorbei.

Mit dem Attribut *clear* (*left*, *right*, *both*) kann ein Umfließen wieder unterbrochen werden. Beispielsweise fließt einen Box mit dem Attribut *clear:left* nicht neben einer Box mit dem Attribut *float:left*, sie wird darunter angezeigt. Siehe auch Abbildung 4.4.

4.5 Textformatierung

Ein wesentlicher Bestandteil der Formatierung von wissenschaftlichen Publikationen ist die Textformatierung [15], [12], daher werden im folgenden die relevanten CSS-Attribute beschrieben.

Die Schriftart eines Elements kann mit dem *font-family* Attribut angegeben werden. Die Angabe mehrerer Schriftarten und Familien ist möglich, falls die erstgenannte Schriftart nicht vorhanden ist wird die nächste genommen. *h1{font-family: „Helvetica“, „Arial“, „sans-serif“}* beispielsweise legt fest das Überschriften erster Ordnung in „Helvetica“ dargestellt werden sollen, falls nicht vorhanden in „Arial“ und als letzte Möglichkeit als irgendeine serifenlose Schrift.

Die Schriftgröße kann mit *font-size* angegeben werden, die Schriftdicke mit *font-weight*. Mit *font-variant* kann angegeben werden ob der Text normal (*normal*) oder in Kapitälchen (*small-caps*) dargestellt werden soll.

Der Schriftstil wird mit *font-style* festgelegt, mögliche Werte sind *normal* (*normal*), *kursiv* (*italic*) und *schräg* (*oblique*).

Die Zeilenhöhe, und damit auch der Zeilenabstand, lässt sich mit *line-height* angeben, Wortabstand mit *word-spacing* und der Buchstabenabstand mit *letter-spacing*.

Die Ausrichtung des Textes kann mit *text-align* eingestellt werden, die relevanten Attribute sind Blocksatz (*justify*), zentriert (*center*), links- (*left*) und rechtsbündig (*right*).

Das Einrücken der ersten Zeile eines Absatzes ist mit *text-indent* möglich.

4.6 Listenformatierung

Die Darstellung von Listen [19], insbesondere der Aufzählungszeichen, kann in CSS mit dem Attribut *list-style-type* angegeben werden.

Es sind neben verschiedenen Aufzählungszeichen, wie zum Beispiel Kreis (*circle*) oder Vier-eck (*square*) auch zahlreiche Nummerierungsarten möglich. Unter anderem mit dezimalen Zahlen (*decimal*), wahlweise auch mit führender Null (*decimal-leading-zero*), große oder kleine römische Zahlen (*upper- / lower-roman*) oder auch große und kleine lateinische Buchstaben (*upper-, lower-latin*) möglich.

Alternativ kann mit *list-style-image* auch ein eigenes Aufzählungszeichen eingebunden werden.

Mit *list-style-position* kann festgelegt werden ob das Aufzählungszeichen innerhalb der Box (*inside*) dargestellt werden soll oder nicht (*outside*).

4.7 Layouts für verschiedene Ausgabemedien

Mit CSS 2.1 ist es möglich medienspezifische Formatierungen festzulegen. Medientypen sind unter anderem *handheld*, *print*, *projection* oder *screen*.

Medienspezifische Bereiche werden im Stylesheet mit *@media typ* eingeleitet. *@media screen{* {font-family: „sans-serif“}}* legt beispielsweise fest das für die Ausgabe auf den Bildschirm eine serifenlose Schrift verwendet werden soll.

In XHTML-Dokumente können alternativ auch, mit dem Attribut *media=“type“* ausgezeichnet, medienspezifische Stylesheets eingebunden werden.

Mit den in CSS 3 neu eingeführten Medienabfragen [26] werden die Auswahlmöglichkeiten um Medieneigenschaften erweitert. Medienabfragen bestehen aus einem Medientyp und keiner oder mehreren Medieneigenschaften. Verknüpfung erfolgt durch die logischen Operatoren „und“ (*and*), „oder“ (*or*) und „nicht“ (*not*). Eine Deklaration für Farbmonitore wäre beispielsweise *@media screen and (color) {...}*.

Die vielfältigen Medieneigenschaften wie zum Beispiel Höhe des Ausgabemediums (*device-height*), Ausrichtung (*orientation*), Seitenverhältniss (*aspect-ratio*) oder Auflösung (*resolution*) lassen sich auch meistens als Mindest- (vorgestelltes *min-*) oder Höchstwert (vorgestelltes *max-*) angeben.

4.8 Automatisch erzeugte und veränderte Inhalte

Für die Formatierung von wissenschaftlichen Publikationen ist auch das automatische Erzeugen oder Verändern von Inhalten erforderlich [18]. Arbeiten wie die Nummerierung von Überschriften oder das Anordnen von Bildern oder ähnliches muss automatisch erfolgen.

Der Zugriff auf automatisch erzeugte Inhalte erfolgt über Pseudoelemente, welche bereits in 4.3 behandelt wurden.

Der Inhalt von (Pseudo)Elementen, Seitenrändern und bestimmter Seitenbereiche (siehe auch 4.9) kann mit dem Attribut *content* verändert werden.

Hierfür stehen eine Vielzahl möglicher automatisch erzeugter Inhalte zur Verfügung, wie unter anderem Anführungszeichen (*quotes*), Zähler (*counter*), URLs (*url*), Text (*string*) sowie der eigentliche Elementinhalt (*contents*).

Kombinationen sind durch Aneinanderreihung möglich, Alternativen werden mit einem Komma getrennt. Beispielsweise zeigt *h1{content: url(1), url(2), contents;}* den Inhalt von Url1, falls vorhanden, ansonsten den Inhalt von Url2 und als letzte Alternative den Elementinhalt von *<h1>*.

Die Darstellung von (Pseudo)Elemente kann verschoben werden, dafür ist das Attribut *move-to* zuständig. Elemente werden aus dem Elementfluss entnommen und an der gewünschten Stelle dargestellt. Mögliche Werte sind *normal*, *here* oder ein Selektor zur Angabe der gewünschten Position.

Die Art der Anführungszeichen kann mit dem Attribut *quotes* festgelegt werden. So lassen sich zum Beispiel verschiedene Anführungszeichen für unterschiedliche Sprachen festlegen. Automatisch erzeugt und eingefügt werden sie mit dem bereits beschriebenen *content* Attribut.

Ein automatisches einschließen von Zitaten kann beispielsweise wie folgt angegeben werden: *blockquote:before {content: open-quote} blockquote:after {content: close-quote}*.

Für die automatische Nummerierung sind die Attribute *counter-increment* und *counter-reset* zuständig.

Mit *counter(name)* wird ein Zähler erzeugt und dargestellt, mit *counter-increment* hochgezählt und mit *counter-reset* zurückgesetzt.

Folgendes Beispiel zeigt eine Möglichkeit Überschriften zweiter Ordnung zu nummerieren: *h2:before {content: counter(section); counter-increment: section;} h1 {counter-reset: section;}*.

Es gibt auch Elemente wie nummerierte Listen, deren Zähler nicht extra angegeben und manipuliert werden müssen. Wird so ein Zähler verschachtelt, also sowohl im Eltern- als auch im Kindelement erhöht, so wird automatisch eine neue Instanz erzeugt. Daher sind auch verschachtelte Listen möglich.

Alle von Listen (siehe 4.6) bekannte Typen sind auch für andere Zähler möglich und können nach dem Zählernamen angegeben werden. Beispielsweise erzeugt *counter(name, upper-latin)* einen Zähler „name“ der in lateinischen Großbuchstaben dargestellt wird.

4.9 Layout für seitenbasierende Medien

Für die Formatierung von zum Druck bestimmter Medien ist ein seitenbasierendes Layout [27] nötig. Seiten besitzen formabhlängige Eigenschaften und werden in verschiedene Bereiche unterteilt. Die Verteilung des Inhaltes erfolgt nach festen Regeln. Im folgenden werden die Möglichkeiten der Formatierung des Inhaltes von Dokumenten für die Darstellung auf seitenbasierenden Medien mit CSS dargestellt.

Hierfür existieren den in 4.4 beschriebenen Elementboxen sehr ähnliche Seitenboxen. Sie bestehen aus dem Seitenbereich, Innenabstand der Seite, Seitenrahmen und dem Seitenrand, welcher in Randboxen für Kopf und Fußzeilen unterteilt ist. Der Aufbau ist in Abbildung 4.5. dargestellt.

Für diese Bestandteile von Seitenboxen können, wie bei normalen Elementboxen auch, verschiedenen Attribute angegeben werden. Darstellungseigenschaften für seitenbasierende Medien werden im Stylesheet mit *@page...* ausgezeichnet. Die Randbereiche können auf ähnliche Weise

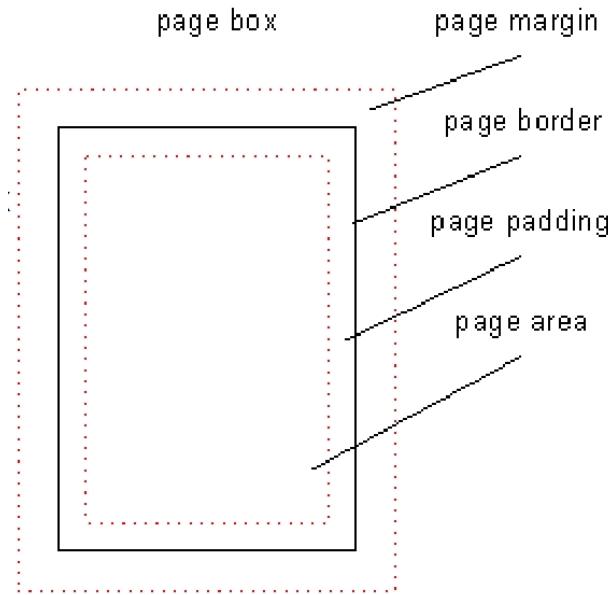


Abbildung 4.5: Aufbau einer CSS Seitenbox. [27]

angesprochen werden, beispielsweise kann mit `@bottom-center{content: „Seite “ counter(page);}` ein Seitenzähler mittig im unteren Seitenrand angebracht werden.

Für seitenbasierende Medien stehen außerdem spezielle Pseudoelemente bereit, welche die Formatierung von linken Seiten (`:left`), rechten Seiten (`:right`) und der erste Seite (`:first`) ermöglichen. Diese erlauben es beispielsweise bei doppelseitig bedruckten Seiten unterschiedliche Ausenränder für die innere und äußere Seite anzugeben.

Regeln für Seitenumbrüche können ebenfalls angegeben werden, und zwar vor (`page-break-before`), nach (`page-break-after`) oder innerhalb (`page-break-inside`) von Blockelementen.

Mögliche Werte für Umbrüche vor oder nach Elementen sind automatisch umbrechen (`auto`), immer umbrechen (`always`), Umbruch vermeiden (`avoid`) oder so umbrechen dass die nächste Seite als linke (`left`) beziehungsweise rechte (`right`) Seite dargestellt wird. Für den Umbruch innerhalb von Elementen sind nur die Angaben automatisch umbrechen und Umbruch vermeiden möglich.

Für die Behandlung von *Schusterjungen* (Seitenumbruch nach der ersten Zeile eines Absatzes) und *Hurenkindern* (Seitenumbruch vor der letzten Zeile eines Absatzes) sind die Attribute `orphans` und `widows` vorhanden. Mit ihnen kann festgelegt werden wie viele Zeilen eines Absatzes mindestens am Ende beziehungsweise Anfang einer Seiten stehen müssen.

4.10 Mehrspaltige Layouts

CSS 3 ermöglicht es auch den Inhalt von Elementen mehrspaltig [23] darzustellen. Die Eigenschaften des mehrspaltigen Layouts können durch die Attribute Spaltenanzahl (`column-count`), Spaltenbreite (`column-width`), Spaltenabstand (`column-gap`) und Spaltenrahmen (`column-rule`) angegeben werden. Für den Spaltenrahmen kann, wie für sonstige Rahmen auch (siehe 4.4), Breite, Farbe und Art angegeben werden.

Spaltenumbrüche vor (`column-break-before`) oder nach (`column-break-after`) Blockelementen können, ähnlich Seitenumbrüchen (siehe 4.9), angegeben werden. Mögliche Werte sind automatischer Umbruch (`auto`) oder Umbruch vermeiden (`avoid`).

Mit dem Attribut `column-span` kann die Anzahl der Spalten über die sich ein Element erstrecken soll, angegeben werden. In folgendem Beispiel erstreckt sich die Überschrift über alle Spalten, siehe Abbildung 4.6.

Des weiteren kann die Art des Spaltenauffüllens (`column-fill`) angegeben werden. Mögliche

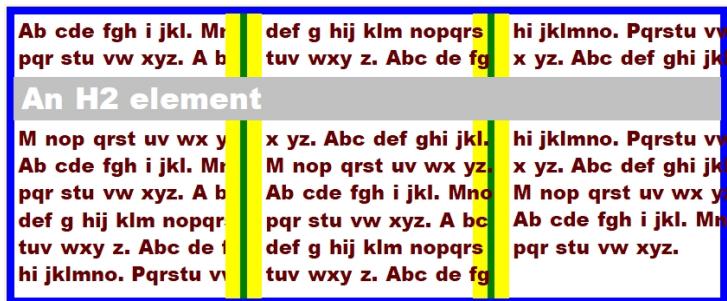


Abbildung 4.6: Mehrspaltiges Layout mit Spaltenübergreifender Überschrift. [23]

Werte sind sequenziell (*auto*), und ausgeglichen (*balance*), was insbesondere bei Spalten fester Höhe wichtig ist. So kann erzwungen werden das mehrere Spalten zur gleichen Höhe aufgefüllt werden.

4.11 Darstellung mathematischer Formeln

Mathematische Formeln können, aufgrund der in 2.2 Beschriebenen Eigenschaften von XHTML, in MathML geschrieben und eingebettet werden.

MathML ist eine auf XML basierende Auszeichnungssprache für mathematische Formeln, in ihr können mathematische Symbole und Ausdrücke ausgezeichnet werden. MathML bietet dafür eine Vielzahl an Elementen, deren Aufbau für die Arbeit nicht relevant ist, und auf die hier nicht weiter eingegangen wird.

Die wichtigsten MathML-Elemente können in Verbindung mit XHTML und CSS genutzt werden. Nicht unterstützt werden hauptsächlich Elemente zur Darstellung welche durch CSS Eigenschaften überflüssig werden.

4.12 CSS Umsetzung

Mit den verfügbaren CSS Eigenschaften werden die in 4.1 und 4.2 beschriebenen Elemente entsprechend den Vorgaben formatiert.

Einen Ausschnitt aus dem CSS für das Format *SIGCHI Conference Proceedings* sehen Sie in Abbildung 4.7.

4.13 Eingeschränkte Browserunterstützung

Im *Mozilla Firefox* ist die Implementierung der @Page Rule (CSS 2.1 siehe 4.9) fehlerhaft [13], daher sind für diesen Browser im Moment keine Angaben für die Seitenränder möglich.

Die Unterstützung von CSS 3 Modulen in den Browsern ist noch eingeschränkt, insbesondere die Module für Seitenbasierende Medien (siehe 4.9) und Mehrspaltige Layouts (siehe 4.10). Daher muss deren Funktionalität durch JavaScript nachgebildet werden, was im Abschnitt 5.2 näher beschrieben wird. Eine Übersicht über die verfügbaren CSS3 Funktionen bietet auch Tabelle 4.2.

```

@page {
    margin-top: 1.905cm; margin-bottom: 2.5cm; padding: 0;
}

@media print, screen {
    body {
        width: 17.78cm; margin:auto; text-align: justify; font-family: "Times New Roman"; font-size: 10pt;
        h1, h2, h3, h4 {
            font-family: "Helvetica", "Arial";
            font-weight: bold;
        }
        h1, h2, h3, h4 {
            margin-top:9pt; margin-bottom:0; text-align: left; font-size: 9pt;
        }
        h1 {
            margin-bottom: 9pt; margin-top: 18pt; text-align: center; font-size: 18pt;
        }
        h2 {
            text-transform: uppercase;
        }
        h4 {
            font-style: italic;
        }
        h2:first-child, h3:first-child, h4:first-child {margin-top:0;}
        p {
            margin-top:0; margin-bottom:0;
        }
        div p + p {
            page-break-inside:avoid; margin:0; padding: 0;
        }
        div {
            margin-top:0; padding-top:0; padding-left: 0; list-style-type: none;
        }
        ol {
            margin-top:0; padding-top:0; padding-left: 0;
        }
        ul {
            margin-top:0; padding-top:0; padding-left: 0;
        }
        li {
            margin-left: 0.425cm; margin-bottom: 8pt;
        }
        hr {
            width: 5cm; border-style: solid; border-width: 0.02cm; text-align: left;
        }
        code {
            text-align : left;
        }
        table {
            border-collapse : collapse;
        }
        th {
            font-weight: bold; text-align: center; border-width : 2px; border-style: solid; padding: 4px;
        }
        td {
            text-align: center; border-width : 2px; border-style: solid; padding: 4px;
        }
        cite {
            font-style: normal;
        }
        .author {
            font-weight: bold;
        }
        .author: first-line {
            font-weight: bold; text-align: center; margin-bottom: 10pt; margin-top: 10pt; font-weight: bold; text-align: center;
        }
        .figCaption {
            font-weight: bold; text-align: center; content: "Figure " counter(figCounter) " "; counter-increment: figCounter;
        }
        .figCaption:before {
            font-weight: bold; margin-top: 10pt; margin-bottom: 10pt; font-weight: bold; text-align: center;
        }
        .tabCaption {
            font-weight: bold; text-align: center; content: "Table " counter(tabCounter) " "; counter-increment: tabCounter;
        }
        .tabCaption:before {
            font-weight: bold; text-align: center; content: counter(tabCounter) " "; counter-increment: tabCounter;
        }
        .footnote:before {
            vertical-align: super; font-size: 8pt; content: counter(footCounter) " "; counter-increment: footCounter;
        }
        .copyrightBox {
            font-size: 8pt;
        }
    }
}

```

Abbildung 4.7: Ausschnitt des CSS für SIGCHI Conference Proceedings.

Funktion:	Umsetzung möglich mit:	Tatsächliche Umsetzung:
Elementanordnung	CSS	CSS
Textformatierung	CSS	CSS
Listenformatierung	CSS	CSS und JS
Layouts für verschiedene Ausgabemedien	CSS	CSS
Automatisch erzeugte und veränderte Inhalte	CSS	CSS und JS
Layout für seitenbasierende Medien	CSS	CSS und JS
Mehrspaltige Layouts	CSS	CSS und JS

Tabelle 4.2: Übersicht über die möglichen CSS3 Funktionen und ihre tatsächliche Umsetzung.

5 JavaScript

Im folgenden werden die wichtigsten Funktionen zur Dokumentveränderung mit JavaScript dargestellt. Anschließend werden Alternativen zu noch nicht verfügbaren CSS Eigenschaften aufgezeigt. Außerdem werden JavaScript Lösungen für in Abschnitt 3 ermittelte Anforderungen, die nichts mit der Darstellung zu tun haben, vorgestellt. Abschließend folgt eine Zusammenfassung in der auf die Möglichkeiten und Limitierungen, insbesondere in zur Zeit verfügbaren Browsern, eingegangen wird.

5.1 Veränderungen des Dokumentbaumes

Die Struktur eines XHTML Dokumentes wird durch einen Dokumentbaum dargestellt, der mit JavaScript verändert werden kann [20], [36]. Dafür gibt es zahlreiche Funktionen, wovon die für die Formatierung von wissenschaftlichen Publikationen wichtigsten im folgenden dargestellt werden.

Die Knoten des Dokumentbaumes werden in Elementen und Textknoten unterteilt, letztere können selbst keine Kinder besitzen und enthalten Textinhalt. Ein Beispiel für den Dokumentbaum einer Tabelle ist in Abbildung 5.1 dargestellt.

Auf das gesamte XHTML Dokument kann durch das Objekt `document` zugegriffen werden, auf den Rumpf durch `document.body` und auf die zugeordneten Stylesheets mit `document.styleSheets[int]`.

Für die Auswahl von Elementen stehen die Funktionen `.getElementById(„id“)` für Elemente denen eine Identität zugewiesen wurde, `.getElementsByClassName(„class“)` für alle Elemente einer Klasse sowie `.getElementsByTagName(„tag“)` für alle Elemente eines Typs zur Verfügung.

Zur Auswahl des Elternelementes gibt es `.parentNode`, für alle Kindknoten `.childNodes`. Die Anzahl der Kindelemente kann mit `.childElementCount` bestimmt werden, mit `.hasChildNodes` kann überprüft werden ob das Element leer ist.

Zur Navigation durch Kind- und Geschwisterknoten gibt es sowohl Funktionen nur für Elemente als auch Funktionen für Elemente und Textknoten. Die folgenden Funktionen liefern nur Elemente, durch weglassen des mittigen „Element“ werden alle Knoten zurückgegeben.

Auf den ersten Kindknoten kann man mit `.firstElementChild` zugreifen, auf den letzten Kindknoten mit `.lastElementChild`. Zur Auswahl des vorherigen Geschwisterknoten gibt es `.previousElementSibling`, für den nachfolgenden `.nextElementSibling`.

Elemente können mit `.createElement(nodeName)` erzeugt werden, Textknoten mit `.createTextNode(string)`. Knoten können mit `.cloneNode(node)` kopiert werden, mit `.appendChild(node)` an ein Element angehängt und mit `.removeChild(node)` aus einem Element entfernt werden. Sie können des weiteren mit `.insertBefore(node, refNode)` vor bestimmten Elementen eingefügt werden sowie mit `.replaceChild(node, oldNode)` andere Knoten ersetzen.

Der Typ eines Knotens kann mit `.nodeName` abgefragt werden, die Klasse mit `.className` und der Inhalt alle Textknoten eines Elementes mit `.wholeText`.

CSS Attribute können mit `.style.attribut = „value“` gesetzt werden und mit `document.defaultView.getComputedStyle(node, null).attribut` abgefragt werden.

Allerdings sind diese Werten, insbesondere die Größe der Elemente betreffend, je nach Browser leicht verschieden.

5.2 Mehrspaltiges Drucklayout

Die Darstellung des mehrspaltigen Drucklayouts erfolgt mit Hilfe von verschachtelten `<div>` Elementen, welche mit dem Inhalt gefüllt werden. Die Struktur einer Seite mit zwei Spalten sehen Sie in Abbildung 5.2.

Der Ablauf des Scriptes ist folgendermaßen:

1. Initialisierung.

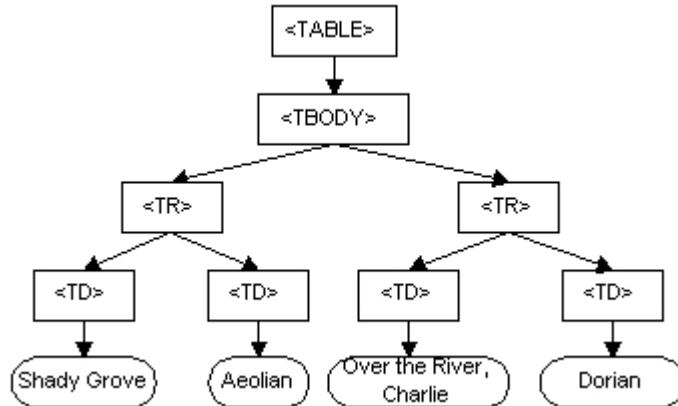


Abbildung 5.1: Darstellung des Dokumentteilbaumes einer Tabelle. [20]

2. BibTeX Datei einlesen.
3. Verweise auflösen.
4. Spalten erstellen und Inhalt einfügen.
5. Zähler erstellen.
6. Inhalte ausrichten.
7. Seitenumbrüche erzeugen.

Bei der Initialisierung wird als erstes ermittelt um welchen Typ von „Paper“ es sich handelt. Dazu wird der Dateiname des Stylesheets (zum Beispiel „sigchi.css“) eingelesen wodurch sich der verwendete Typ („sigchi“) ergibt.

Je nach Typ werden dann die entsprechenden Eigenschaften wie Größe des Inhaltsbereiches, Anzahl und Abstand der Spalten und automatische Elementanordnung festgelegt. Die Höhe der Spalten der ersten Seite wird berechnet, wobei auf die Größe vorhergegangener Elemente und, falls vorhanden, die Größe der Copyright Box eingegangen wird. Es werden die *dpi* (dots per inch, engl. für Punkte pro Zoll) des Monitors sowie die Höhe einer Zeile bestimmt, außerdem wird festgelegt ob es sich um die Bildschirm oder Druckausgabe handelt.

Im nächsten Schritt wird, mit Hilfe des OpenSource *Javascript BibTeX Parsers* [17] die BibTeX-Datei eingelesen, welche als *<object>* im XHTML-Dokument eingebunden sein muss.

Die BibTeX-Einträge werden entsprechend der Formatvorlage des „Papers“ formatiert und in das XHTML-Dokument eingefügt. Für das Ausgabemedium Bildschirm werdenUrls in den Quellenangaben in Hyperlinks umgewandelt, falls keine Url vorhanden ist wird ein Hyperlink auf eine Suchanfrage für Google Scholar erzeugt.

Im nächsten Schritt werden sämtliche mittels *<cite>* gekennzeichnete Verweise auf Bilder, Tabellen, Formeln, Fußnoten oder Literaturangaben aufgelöst. Bei der Druckausgabe werden sie durch der Vorgabe entsprechende Zeichen (zum Beispiel „[1]“) ersetzt, bei der Bildschirmausgabe durch Hyperlinks.

Anschließend werden, für die Druckausgabe, *<div>* Behälter für die Spalten erstellt und mit Inhalt gefüllt. Der Inhalt wird elementweise in die Spalten kopiert, falls ein Textelement zu groß ist wird es zeilenweise eingefügt.

Schusterjungen, das ist die erste Zeile eines Absatzes am Ende einer Spalte, und *Hurenkinder*, das ist die letzte Zeile eines Absatzes am Anfang einer Spalte, sowie alleinstehende Überschriften werden vermieden.

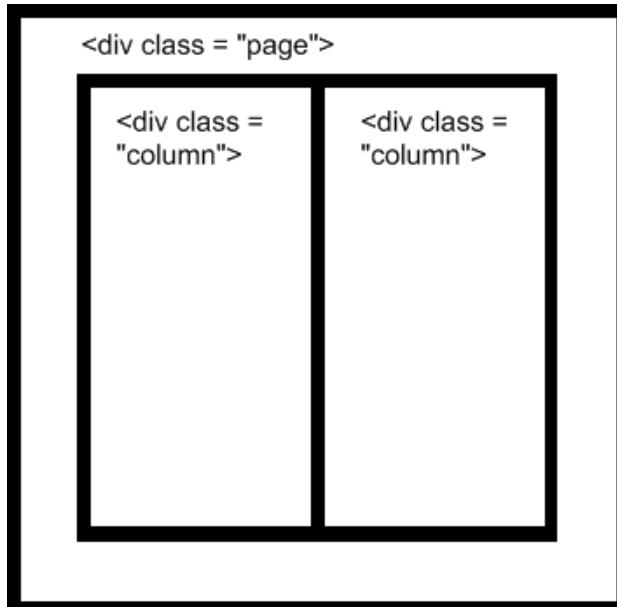


Abbildung 5.2: Seitenaufteilung mit *<div>* Behältern.

Bei ersten Zeilen eines Absatzes und Überschriften wird dazu geprüft ob noch Platz für eine zusätzliche Zeile ist, falls nicht wird vor ihnen umgebrochen. Bei letzten Zeilen eines Absatzes wird geprüft ob sie am Anfang einer Spalte stehen, dann wird von der vorherigen Spalte die letzte Zeile geholt.

Bilder und Tabellen werden nur zusammen mit ihren Überschriften eingefügt, falls dadurch am Spaltenende Platz frei bleibt wird dieser mit nachfolgendem Text gefüllt. Fußnoten werden direkt nach dem Element, in dem das erste mal auf sie verwiesen wird, eingefügt.

Da die CSS eigenen Zähler aufgrund der verschachtelten *<div>* Behälter nicht verwendbar sind (siehe 4.8 und 4.6), werden im nächsten Schritt Zähler für Überschriften und Listen erzeugt und entsprechend der Formatvorgaben eingefügt.

Anschließend folgt das Ausrichten der Inhalte. Für die Druckausgabe werden die Spalten ausgerichtet und die Fußnoten ans Spaltenende verschoben. Für mache „Paper“-Typen ist erforderlich, dass Bilder, Tabellen und Formeln in den Ecken der Spalten platziert werden. Das wird, falls eingestellt, ebenfalls in diesem Schritt vorgenommen.

Bei der Ausgabe auf dem Monitor werden lediglich die Fußnoten ans Ende des Dokumentes geschoben.

Abschließend wird für die Druckausgabe noch für jede Seite ein eigener *<div>* Behälter mit erzwungenem Seitenumbruch erstellt und gefüllt.

6 FAZIT

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
Table 1. Vis Paper Acceptance Rate				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

Table 1. Vis Paper Acceptance Rate

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
Table 1. Meccat Head				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

Table 1. Meccat Head

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
2.1 Mezzat Head				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

2.1 Mezzat Head

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
2.1 Mezzat Head				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

2.1 Mezzat Head

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
Ejector Seat Reservation				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

Ejector Seat Reservation

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
Ejector Seat Reservation				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

Ejector Seat Reservation

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

$\sum_{j=1}^x j = \frac{x(x+1)}{2}$				(1)
Ejector Seat Reservation				
Year	Submitted	Accepted	Accepted (%)	
1994	91	41	45.1	
1995	102	41	40.2	
1996	101	43	42.6	
1997	117	44	37.6	
1998	118	50	42.4	
1999	67	36	53.7	
2000	131	52	34.4	
2001	152	51	33.6	
2002	138	38	27.6	
2003	192	63	32.8	
2004	204	46	27.6	
2005	268	88	33.0	
2006	228	63	27.6	

Ejector Seat Reservation

Year Submitted Accepted Accepted (%)

1994 91 41 45.1
1995 102 41 40.2
1996 101 43 42.6
1997 117 44 37.6
1998 118 50 42.4
1999 67 36 53.7
2000 131 52 34.4
2001 152 51 33.6
2002 138 38 27.6
2003 192 63 32.8
2004 204 46 27.6
2005 268 88 33.0
2006 228 63 27.6

Abbildung 6.1: Gegenüberstellung einer orginalen PDF-Datei (a) einer Seite eines „Paper“-Templates mit der aus XHTML, CSS und JS erzeugten PDF-Datei (b).

6 Fazit

Wie am Anfang der Arbeit (2.3) dargestellt, hat die Formatierung wissenschaftlicher Publikationen mit XHTML und CSS erhebliche Vorteile gegenüber den konventionellen Formaten.

Einsetzbar wird sie allerdings nur wenn die Rückwärtskompatibilität bezüglich des Abgabeformates gegeben ist, also den Formatvorlagen entsprechende PDF-Dateien erzeugt werden können. Diese Anforderungen wurden anhand gängiger Formate untersucht (3) und es wurden Lösungsmöglichkeiten mit CSS 3 dargestellt (4). Anschließend wurde überprüft welche dieser CSS 3 Funktionen bereits verfügbar ist, fehlende Funktionen wurden mit JavaScript (5) nachgebaut. Das Ergebniss sehen Sie in einer Gegenüberstellung einer orginalen PDF-Datei eines „Paper“-Templates mit der aus XHTML, CSS und JS erzeugten PDF-Datei in Abbildung 6.1.

Abschließend kann man sagen das die Formatierung wissenschaftlicher Publikationen mit XHTML und CSS zum jetzigen Zeitpunkt, trotz teilweise fehlender CSS 3 Implementierung, möglich ist.

Mit fortschreitender CSS 3 Implementierung werden die Möglichkeiten für die Formatierung mit XHTML und CSS weiter zunehmen, wodurch sich XHTML und CSS auch als Austauschformat für wissenschaftlicher Publikationen durchsetzen könnte.

Fig. 1. Sample illustration.

*Footnotes appear at the bottom of the column

Fig. 1. Sample illustration

(b)

Inhalt der beigelegten CD

1. Readme-Datei.
2. Quelltext.
3. Projektarbeit im PDF-Format.
4. Projektarbeit im LaTeX-Format.
5. Folien der Endpräsentation.
6. Paper Templates orginale PDF-Dateien.
7. Paper Templates erzeugte PDF-Dateien.

Literatur

- [1] Ben Adida, Mark Birbeck, Shane McCarron, and Steven Pemberton. *RDFa in XHTML: Syntax and Processing*. World Wide Web Consortium, Oktober 2008. Available from: <http://www.w3.org/TR/rdfa-syntax/>.
- [2] Adobe. Adobe pdf (portable document format), 2009. Available from: <http://www.adobe.com/de/products/acrobat/adobepdf.html>.
- [3] Murray Altheim and Shane McCarron. *XHTML 1.1 - Module-based XHTML*. World Wide Web Consortium, Mai 2001. Available from: <http://www.w3.org/TR/xhtml11/>.
- [4] Apple. Safari 4, 2009. Available from: <http://www.apple.com/de/safari/download/>.
- [5] Daniel Austin, Subramanian Peruvemba, Shane McCarron, Masayasu Ishikawa, and Mark Birbeck. *XHTML Modularization 1.1*. World Wide Web Consortium, Oktober 2008. Available from: <http://www.w3.org/TR/xhtml-modularization/>.
- [6] Bert Bos. *CSS basic box model*. World Wide Web Consortium, August 2007. Available from: <http://www.w3.org/TR/css3-box/>.
- [7] Bert Bos. *Cascading Style Sheets Current Work*. World Wide Web Consortium, 2009. Available from: <http://www.w3.org/Style/CSS/current-work/>.
- [8] Bert Bos, David Carlisle, George Chavchanidze, Patrick D. F. Ion, and Bruce R. Miller. *A MathML for CSS profile*. World Wide Web Consortium, June 2009. Available from: <http://www.w3.org/TR/mathml-for-css/>.
- [9] Bert Bos, Tantek Celik, Ian Hickson, and Hakon Wium Lie. *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification*. World Wide Web Consortium, April 2009. Available from: <http://www.w3.org/TR/CSS2/>.
- [10] Ben Caldwell, Michael Cooper, Loretta Guarino Reid, and Gregg Vanderheiden. *Web Content Accessibility Guidelines (WCAG) 2.0*. World Wide Web Consortium, Dezember 2008. Available from: <http://www.w3.org/TR/WCAG20/>.
- [11] Tantek Celik, Elika J. Etemad, Daniel Glazman, Ian Hickson, Peter Linss, and John Williams. *Selectors Level 3*. World Wide Web Consortium, März 2009. Available from: <http://www.w3.org/TR/css3-selectors/>.
- [12] John Daggett. *CSS Fonts Module Level 3*. World Wide Web Consortium, Juni 2009. Available from: <http://www.w3.org/TR/css3-fonts/>.
- [13] Joseph T. Duncan. *Bug 115199 - @page in CSS2 not implemented*. Mozilla Foundation, 2009. Available from: https://bugzilla.mozilla.org/show_bug.cgi?id=115199.
- [14] Brendan Eich. *Standard ECMA-262 ECMAScript Language Specification*. ECMA International, Dezember 1999. Available from: <http://www.ecma-international.org/publications/standards/Ecma-262.htm>.
- [15] Elika J. Etemad and Paul Nelson. *CSS Text Level 3*. World Wide Web Consortium, März 2007. Available from: <http://www.w3.org/TR/css3-text/>.
- [16] Google. Chrome beta, 2009. Available from: <http://www.google.com/chrome/eula.html?extra=betachannel>.

- [17] Steve Hannah. *Javascript BibTeX Parser*, 2009. Available from: <http://sourceforge.net/projects/jsbibtex/>.
- [18] Ian Hickson. *CSS3 Generated and Replaced Content Module*. World Wide Web Consortium, Mai 2003. Available from: <http://www.w3.org/TR/css3-content/>.
- [19] Ian Hickson and Tantek Celik. *CSS3 module: Lists*. World Wide Web Consortium, November 2002. Available from: <http://www.w3.org/TR/css3-lists/>.
- [20] Arnaud Le Hors, Philippe Le Hégaret, Lauren Wood, Gavin Nicol, Jonathan Robie, Mike Champion, and Steve Byrne. *Document Object Model (DOM) Level 3 Core Specification*. World Wide Web Consortium, April 2004. Available from: <http://www.w3.org/TR/DOM-Level-3-Core/>.
- [21] IEEE. Formatting guidelines for vis/ infovis 2009, 2009. Available from: http://www.cs.sfu.ca/~vis/Tasks/camera_tvcg.html.
- [22] Hakon Wium Lie. *Cascading Style Sheets*. PhD thesis, University of Oslo, 2005. Available from: <http://people.opera.com/howcome/2006/phd/>.
- [23] Hakon Wium Lie. *CSS3 module: Multi-column layout*. World Wide Web Consortium, Juni 2007. Available from: <http://www.w3.org/TR/css3-multicol/>.
- [24] Hakon Wium Lie and Bert Bos. *Cascading Style Sheets: Designing for the Web, 3rd Edition*. Addison-Wesley Professional, April 2005. Available from: <http://www.informit.com/store/product.aspx?isbn=0321193121&rll=1>.
- [25] Hakon Wium Lie and Bert Bos. Printing a book with css: Boom! *A List Apart Magazine*, 1(208), 2005. Available from: <http://www.alistapart.com/articles/boom/>.
- [26] Hakon Wium Lie, Tantek Celik, Daniel Glazman, and Anne van Kesteren. *Media Queries*. World Wide Web Consortium, April 2009. Available from: <http://www.w3.org/TR/css3-mediaqueries/>.
- [27] Hakon Wium Lie and Melinda Grant. *CSS3 Module: Paged Media*. World Wide Web Consortium, Oktober 2006. Available from: <http://www.w3.org/TR/css3-page/>.
- [28] Hakon Wium Lie and Janne Saarela. Multipurpose web publishing using html, xml, and css. *Commun. ACM*, 42(10):95–101, 1999. Available from: <http://portal.acm.org/citation.cfm?doid=317665.317681>.
- [29] Luc Maranget. Hevea, 2007. Available from: <http://pauillac.inria.fr/hevea/>.
- [30] Shane McCarron and Ishikawa Masayasu. *XHTML Media Types - Second Edition*. World Wide Web Consortium, Januar 2009. Available from: <http://www.w3.org/TR/xhtml-media-types/>.
- [31] Microformats. Microformats, 2009. Available from: <http://microformats.org>.
- [32] Mozilla. Firefox nightly builds, 2009. Available from: <http://ftp.mozilla.org/pub.mozilla.org/firefox/nightly/latest-trunk/>.
- [33] Opera. Opera snapshot, 2009. Available from: <http://my.opera.com/desktopteam/blog/>.
- [34] Oren Patashnik. *BibTeXing*, Februar 1988. Available from: <http://dante.ctan.org/CTAN/biblio/bibtex/contrib/doc/btxdoc.pdf>.

- [35] Steven Pemberton. *XHTML 1.0 The Extensible HyperText Markup Language (Second Edition)*. World Wide Web Consortium, August 2002. Available from: <http://www.w3.org/TR/xhtml1/>.
- [36] Doug Schepers and Robin Berjon. *Element Traversal Specification*. World Wide Web Consortium, Dezember 2008. Available from: <http://www.w3.org/TR/ElementTraversal/>.
- [37] SigChi. Chi formatting instructions, 2009. Available from: <http://www.chi2009.org/Authors/Guides/Formatting.html>.
- [38] Springer. Information for lncs authors, 2009. Available from: <http://www.springer.com/computer/lncs?SGWID=0-164-7-72376-0>.
- [39] Webkit. Webkit nightly builds, 2009. Available from: <http://nightly.webkit.org/>.
- [40] YesLogic. Prince xml, 2009. Available from: <http://www.princexml.com/>.