Kai Laborenz

CSS-Praxis

Galileo Computing

Liebe Leserin, lieber Leser,

Seit der ersten Auflage dieses Buches ist gerade ein Jahr vergangen. CSS-Design hat seitdem Einzug gehalten in viele Bereiche des modernen Webdesigns. Selbst große Webprojekte, wie der Internetauftritt des Sterns oder der deutschen Post, haben seit Ende 2002 einen Relaunch in CSS hinter sich. Unaufhaltsam setzt sich CSS-Design in der alltäglichen Praxis durch und verdrängt das althergebrachte Tabellenlayout der reinen HTML-Welt.

Wenn Sie bisher noch wenig oder gar keine Cascading Stylesheets eingesetzt haben, können Sie mit dem Buch von Kai Laborenz lernen, welche vielen neuen Möglichkeiten sich für Ihre Internetseite durch den Einsatz von CSS ergeben. Endlich macht Webdesign wieder Spaß. Darüber hinaus bieten Cascading Stylesheets viele Möglichkeiten, Seiten nicht nur für den Einsatz auf dem PC, sondern auch auf anderen Geräten wie PDA oder Handy anzupassen – Machen Sie Ihre Webseiten zukunftssicher.

Diejenigen, die bereits CSS einsetzen, finden in diesem Buch viele Tipps und Tricks aus der Praxis. Hier hilft Ihnen der Autor dabei, browserübergreifende Lösungen zu entwickeln. Die herausnehmbare Referenzkarte wird eine unentbehrliche Hilfe bei Ihrer täglichen Coding-Arbeit werden. Auf einen Blick sehen Sie, welche Browser welche CSS-Elemente unterstützen. Und – schauen Sie doch mal auf der Webseite des Autors www.css-praxis.de vorbei. Hier finden Sie Hilfestellung im Forum, neue Entwicklungen und Buch-Updates.

Ich freue mich, wenn Sie uns Ihre kritischen Anmerkungen, Ihr Lob oder Ihre Verbesserungsvorschläge zukommen lassen möchten.

Stephan Mattescheck
Lektorat Galileo Computing

stephan.mattescheck@galileo-press.de
www.galileocomputing.de

Galileo Press • Gartenstraße 24 • 53229 Bonn

Auf einen Blick

Inhalt

Vorwort zur zweiten Auflage

Ein Jahr ist schnell vergangen – besonders im Web. Trotzdem ist es schon erstaunlich, wie viel sich auf dem Gebiet der Stylesheets seit der ersten Ausgabe, die vor ziemlich genau einem Jahr erschienen ist, getan hat.

War es damals noch ziemlich schwierig, bedeutendere Websites zu finden, die CSS nutzten, so herrscht inzwischen auch in Deutschland kein Mangel mehr an guten Beispielen. Ein Grund für die stetig steigende Verbreitung von CSS-Sites ist sicher das langsame Aussterben der »CSS-feindlichen« Browser – der Marktanteil von Netscape 4 ist im letzten Jahr fast um die Hälfte gefallen und bewegt sich inzwischen im einstelligen Bereich. Auch das wachsende Bewusstsein, dass Barrierefreiheit und verbindliche Standards für die Zukunft des Webdesigns wichtig sind, hat zur Verbreitung beigetragen.

Während ich im Vorwort zur ersten Auflage noch zu mutigem Voranschreiten aufgefordert habe, ist CSS inzwischen auf dem besten Weg, Webdesign-Alltag zu werden. In diesem Sinne wünsche ich Ihnen beim Lesen des Buches und beim Ausprobieren der Beispiele und Anleitungen viel Spaß am (noch nicht ganz) Alltäglichen.

Was erwartet Sie in diesem Buch?

Dieses Buch will mit einem praxisbezogenen Ansatz die Möglichkeiten, Schwierigkeiten und Begrenzungen von Cascading Stylesheets anhand konkreter Beispiele aufzeigen.

In den ersten beiden Kapiteln gebe ich eine Einführung zum Aufbau und den grundlegenden Konzepten von Cascading Stylesheets und zeige ein einfaches Beispiel.

In den Kapiteln 3 bis 5 geht es um die praktische Umsetzung: Wie sieht die Entwicklung von CSS aus, wie reagieren die verschiedenen Browser auf CSS und was gibt es beim Praxiseinsatz zu beachten?

Die Kapitel 6 und 7 sind Beispielen und konkreten Skripten gewidmet – zum Nachvollziehen und zum praktischen Einsatz.

Kapitel 8 zeigt Ihnen eine Reihe von Programmen, die beim Erstellen von Stylesheets nützlich sind.

Und schließlich Kapitel 9, das – last but not least – eine vollständige Referenz aller CSS-Eigenschaften mit kompletter Beschreibung, beispielhafter Syntax und Anmerkungen zur Browserkompatibilität enthält.

Auf der beiliegenden CD-ROM finden Sie alle Beispiele und Abbildungen des Buches. Außerdem alle besprochenen Skripte und Programme als Freeware, Shareware oder Demoversionen, darunter auch die beiden Website-Editoren Dreamweaver und GoLive in der aktuellen Version. Die Screenshots, die im Buch abgebildet sind, finden Sie auf der CD-ROM in Farbe.

Stylesheet-Tabelle zum Herausnehmen

Zusätzlich liegt diesem Buch eine Stylesheet-Kompatibilitätstabelle bei. Dort sind in Kurzform alle wichtigen Eigenschaften von Stylesheets und ihre Unterstützung durch die wichtigsten Browser aufgelistet. Die Tabelle können Sie beim Arbeiten zur schnellen Referenz benutzen und für Details dann im Buch nachlesen.

Wie sollten Sie dieses Buch lesen?

Sie können das Buch natürlich ganz normal von vorn nach hinten durchlesen. Ich empfehle das vor allem, wenn Sie mit Cascading Stylesheets bisher wenig zu tun hatten. Wenn Sie schon etwas über CSS wissen, können Sie auch die Praxisbeispiele ab Kapitel 5, *CSS in der Praxis*, durchgehen und nur im Zweifelsfall die Referenzen des ersten Teils zu Rate ziehen. Oder Sie fangen gleich mit den Stylesheet-Beispielen in Kapitel 6, *Skripte*, an und sehen, was Sie für Ihre Website anwenden können.

Icons und Hinweise

Um Sie auf bestimmte Dinge aufmerksam zu machen, verwenden wir ein paar Icons in der äußeren Spalte:

Hinweis: Ergänzende Hinweise zum Thema

Falle: Browserfehler oder andere Fallen für CSS-Designer

Warnung: Warnungen oder Achtung, wichtiger Hinweis

Tool: Werkzeuge für den CSS-Entwickler

Beispiele: Skripte und Quellcodelistings

CD-ROM: Ergänzendes Material oder Skripte zum Verwenden auf der beiliegenden CD-ROM

Vielen Dank!

Bedanken möchte ich mich bei den vielen Menschen, die mich bei der Arbeit an diesem Buch unterstützt und inspiriert haben. Dazu gehören all diejenigen, die mir Teile ihrer eigenen Arbeiten für dieses Buch zur Verfügung gestellt haben wie Michael Kaspar, Mark Howells, Bjoern Hoehrmann, Tomas Caspers, Daniel Ludwin, Ernesto Gimenez, Winfried Schoech, den e-workers und David Andersson (aka Liorean). Außerdem den Lesern der Mailinglisten **css-design** von Michael Charlier und **i-worker** von Erwin Forner, mit denen ich manches Problem diskutieren konnte.

Vielen Dank an meinen Lektor Stephan Matteschеck, der mich nicht nur auf die Idee zu diesem Buch brachte, sondern mich auch bei der Erstellung unterstützte und insbesondere viel Geduld mit mir hatte, was das verspätete Abgeben von Manuskripten betraf ...

Dank gebührt auch diesen Menschen für ihre wegweisenden Veröffentlichungen zum Thema: Eric Meyer, Jeffrey Zeldman, Eric Costello, Owen Briggs (vor allem für seine vielen Screenshots) und Stefan Münz für SelfHTML.

Vor allem danke ich meinen Kollegen und Mitarbeitern bei Sunbeam und meiner Freundin, die während meiner Arbeit an diesem Buch unter meiner Doppelbelastung leiden mussten.

Kai Laborenz
Berlin, November 2003

1 Einführung in CSS

1 Einführung in CSS

Robert seufzte, als er auf die Uhr blickte. Schon halb neun, und er würde sicher noch einige Stunden benötigen, um die Aufgabe zu erledigen, die ihm sein Abteilungsleiter Tom am Nachmittag mit den Worten »Nur eine Kleinigkeit!« übergeben hatte.

Sicher – genau genommen waren es nur Kleinigkeiten, die auf der Website ihres größten Kunden, der Tempelhofer Werkzeugmaschinen AG – kurz TWAG –, zu ändern waren: Die Initialen sollten einen anderen Schrifttyp bekommen und die Zeilenhöhe der Texte sollte um eine Kleinigkeit verändert werden. Außerdem sollten ein paar Überschriften eine andere Farbe erhalten.

Leider hatte Robert bei der Erstellung der Website alle Initialen als kleine Grafiken angelegt und auch den Zeilenabstand mit Hilfe von vielen kleinen Grafiken als »Abstandshalter« erzeugt. Damals war ihm das noch als sehr gute Idee vorgekommen und er war auch ziemlich stolz gewesen, wie gut »seine« Website in das Corporate Design der TWAG passte. Wie hatte er damals nur vergessen können, an Änderungen zu denken?

Jetzt würde er alle Initialen als Grafiken neu anlegen und in den über hundert Seiten ersetzen müssen. Außerdem würde er die Abstandshalter für die Zeilenhöhe neu anlegen und Initialen für die neu hinzukommenden Textblöcke anlegen müssen. Und er müsste auf jeder Seite die Schriftfarbe der Überschriften ändern.

Irgendwie müsste das doch einfacher gehen ... Robert seufzte erneut und wandte sich wieder dem Monitor zu.

1.1 Von HTML zu CSS

Als Tim Berners-Lee im Jahr 1990 die Sprache HTML entwickelte, dachte er nicht an das Aussehen von Dokumenten. Ihm ging es um eine Sprache, mit der sich Struktur und Inhalte einer Seite beschreiben lassen. HTML-Dokumente bestehen aus einer Ansammlung von ineinander verschachtelten Elementen, die eine hierarchische Struktur (ähnlich der Ordnerstruktur eines Dateisystems) darstellen. Das erste und oberste Element einer jeden HTML-Seite ist das HTML-Element. Darin befinden sich die Elemente HEAD und BODY, die wiederum ihrerseits Elemente enthalten (z.B. Absätze oder Bilder im BODY-Element). So finden sich im Sprachschatz der ersten HTML-Versionen auch nur Befehle, welche die Funktion oder Bedeutung eines Dokumentenbestandteils beschreiben:

- `<p> ... </p>` beschreibt einen Abschnitt,
- `<h1> ... </h1>` eine Überschrift erster Ordnung (die wichtigste),
- `<a href="..."> ... </a>` einen Link auf ein anderes Dokument usw.

Strukturierte Dokumente mit HTML

Die Idee dieser Konstruktion war es, ein Dokument so zu beschreiben, dass es auf vielen unterschiedlichen Anzeigegeräten gemäß seiner Bedeutung wiedergegeben werden kann (»logische Auszeichnung«). So ist es für ein HTML-Dokument im Prinzip egal, ob es auf einem hoch auflösenden 19''-Monitor oder einem monochromen Handydisplay angezeigt werden soll. Das jeweilige Endgerät entscheidet selbst, wie z.B. eine Überschrift 1. Ordnung aussehen soll.

In »reinem« HTML erstellte Webseiten sind dann zwar sehr gut strukturiert, sehen aber auch wenig ansprechend aus.

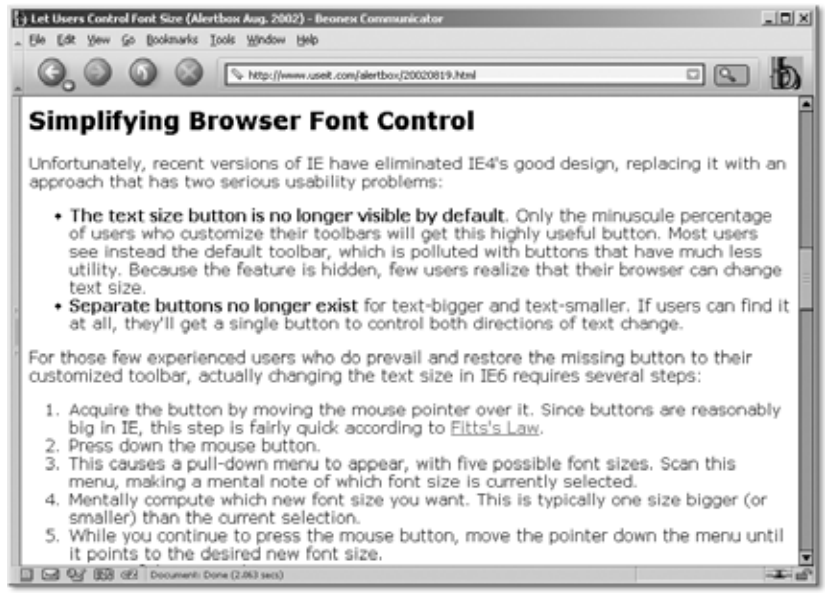

Abbildung 1.1 »Nur-HTML«-Websites beschränken sich auf die Darstellung der Struktur eines Dokuments.

Das in Deutschland wohl bekannteste und beliebteste Kompendium zu HTML ist SelfHTML von Stefan Münz. Das inzwischen in der Version 8 erschienene elektronische Dokument ist nicht nur eine gute Einführung und ein unverzichtbares Nachschlagewerk für HTML, sondern bietet auch Informationen zu XML, CSS, JavaScript und anderen Programmiersprachen, die für Webentwickler interessant sind.

Ergänzt wird es durch allgemeine Hinweise über das Web-Publizieren. Sie finden die aktuelle Version auf der beiliegenden CD-ROM. Unter **http://selfaktuell.teamone.de** existiert ein Online-Portal mit News, Tipps und Diskussionsforen rund um HTML.

Je mehr das WWW an Bedeutung gewann und aus dem Wirkungsbereich der Universitäten heraustrat, desto stärker wurden Wünsche, das Erscheinungsbild von Webseiten genauer bestimmen zu können. Die neueren Browser begannen dann, an der Sprachdefinition von HTML vorbei, eigene Befehle zu interpretieren. Der damals marktbeherrschende Netscape Navigator führte z.B. mit der Version 2 die Möglichkeit ein, Schriftarten, -größen und -farben zu definieren.

Pixelgenaue Layouts durch HTML-Tricks

Mit den Forderungen nach pixelgenauen Layouts konfrontiert, erkannten findige Webdesigner schnell, dass sich viele HTML-Befehle auch zu Layoutzwecken nutzen lassen. Gerade die Tabellenfunktion lässt sich – wenn die Tabellenränder abgeschaltet werden – ausgezeichnet zum Herstellen von Gestaltungsrastern verwenden, wie sie in gedruckten Publikationen üblich sind. Das sieht z.B. so aus:

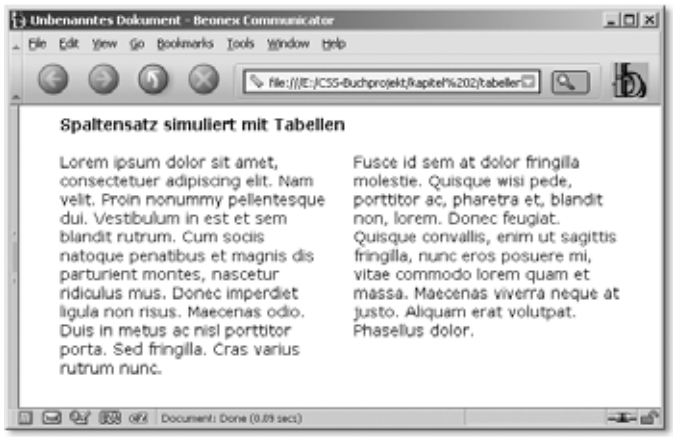

Abbildung 1.2 Zweispalten-Layout mit unsichtbarer Tabelle

Der Trick wird sichtbar, wenn wir die Ränder der Tabelle einschalten (siehe Abbildung 1.3)

Speziell große Websites mit komplexen Layouts setzen eine Konstruktion von mehrfach ineinander verschachtelten Tabellen ein, um ein möglichst exaktes Layout zu erreichen (hier die Homepage des Magazins »Spiegel«).

Abbildung 1.3 Durch Einschalten der Ränder wird die Tabellenkonstruktion enttarnt.

Abbildung 1.4 Der beliebte Online-Dienst des Spiegel setzt beim Design auf Tabellenkonstruktionen ...

Abbildung 1.5 ... was deutlich wird, wenn wir die Ränder der Tabellen sichtbar machen (schwarze Linien).

Nachteile des Tabellenlayouts

Das funktioniert zwar, hat aber eine Reihe von Nachteilen. Zum einen wird der HTML-Code der Webseiten unnötig aufgebläht und kompliziert, was nicht nur die Ladezeit verlängert. Wenn Browser eine normale HTML-Seite schon teilweise anzeigen, während die letzten Reste noch geladen werden, so wird eine Tabelle erst dann dargestellt, wenn alle darin befindlichen Bestandteile vollständig geladen sind. Auch die Wartung von Webseiten wird dadurch sehr schwierig. Zum anderen geht die ursprüngliche Idee der plattformunabhängigen Darstellung dabei verloren.

Da die Vielfalt der möglichen Endgeräte (Computer-Monitore, Fernsehgeräte, Organizer, Handys, Screenreader) immer weiter zunimmt, und selbst auf PCs die Unterschiede zwischen den Betriebssystemen, Webbrowsern und sogar unterschiedlichen Versionen desselben Browsers enorm sind, stößt das »Tricksen« schnell an seine Grenzen. Entweder werden immer größere Gruppen von Benutzern einfach ignoriert ...

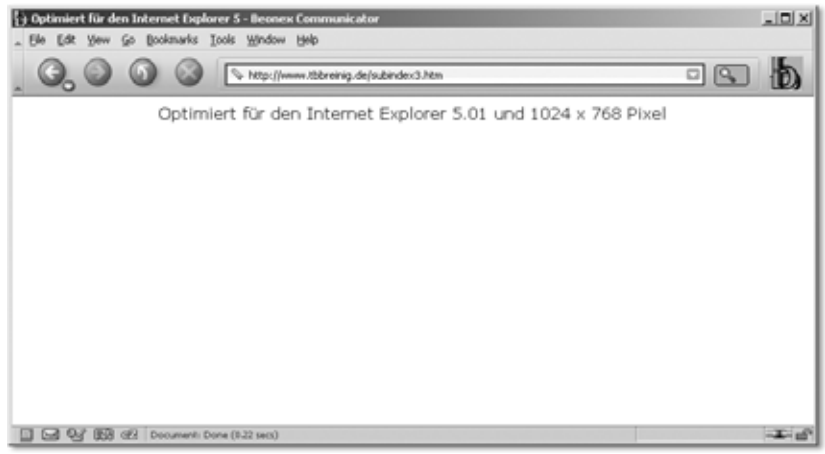

Abbildung 1.6 So geht's natürlich auch – wer nicht den richtigen Browser hat, muss draußen bleiben!

... oder der Entwicklungsaufwand für verschiedene Versionen derselben Website steigt ins Absurde. Durch die Vermischung von Inhalt und Layoutinformationen sind abwärtskompatible Seiten kaum möglich und nachträgliche Änderungen des Aussehens mühsam und kompliziert.

Tabellen sind nicht barrierefrei

Auch sehbehinderte Nutzer, die sich eine Website vorlesen lassen müssen, stoßen bei solchen Websites auf erhebliche Hürden. So lesen Screenreader beispielsweise den Bildschirminhalt einfach zeilenweise vor – was bei unserem zweispaltigen Layout vom Anfang sehr verwirrend wird.

1.2 Was sind CSS?

Cascading Stylesheets sind die Antwort auf die widersprüchlichen Anforderungen nach möglichst genauer Kontrolle über das Aussehen von Websites einerseits und nach »Säuberung« des HTML-Codes von Formatierungs- und Layoutanweisungen andererseits.

Die wesentliche Idee von Cascading Stylesheets ist es, den HTML-Code einer Website von allen Formatierungsbefehlen zu befreien. Diese werden getrennt notiert – in einem Stylesheet.

So wird die konsequente Trennung von Struktur, Aussehen und Inhalten gewahrt:

- HTML-Befehle sind nur für die logische Beschreibung eines Dokuments zuständig.
- Zugeordnete Stil-Anweisungen sorgen für die Formatierung und das Aussehen des Dokuments.
- Die Inhalte stehen als normaler Text zwischen den HTML-Tags.

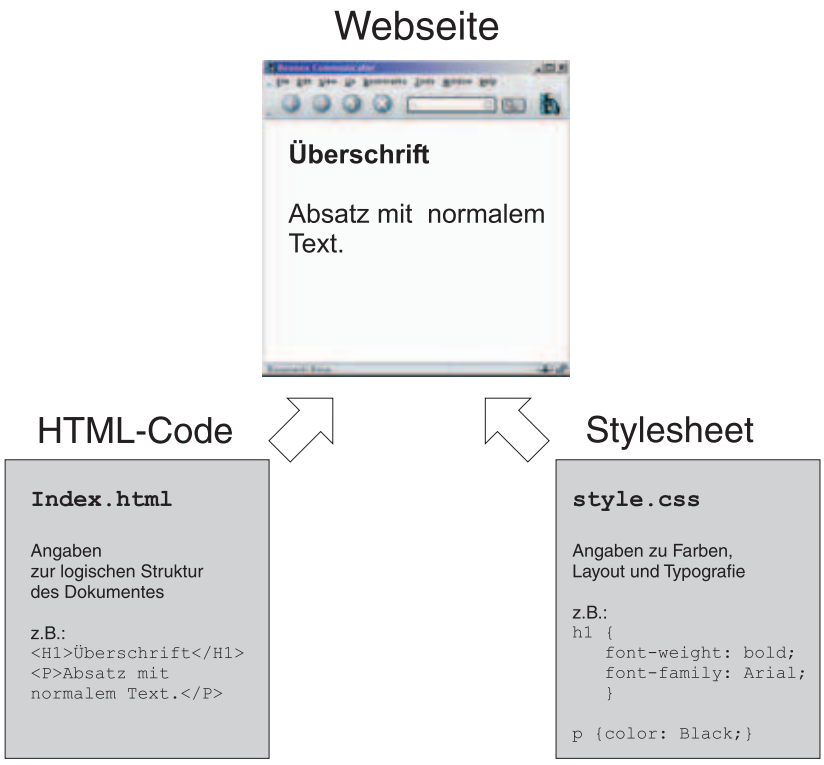

Abbildung 1.7 Aus HTML-Code und Stylesheet entsteht eine Webseite.

Einem HTML-Dokument kann durch Änderung des Stylesheets ein anderes Aussehen gegeben werden. Für unterschiedliche Endgeräte können unterschiedliche Stylesheets bereitgestellt werden, und Benutzer können sogar eigene Stylesheets verwenden, beispielsweise um bei Sehschwäche die Schriftgröße heraufzusetzen.

Erweiterte Formatierungsmöglichkeiten

Neben dieser Trennung von Struktur und Gestaltung erweitern Stylesheets die Möglichkeiten gewaltig, Elemente eines HTML-Dokumentes zu formatieren. Neben Schriftart, -größe und -farbe, die auch mit dem HTML-Tag `<font>` beeinflusst werden können, erlaubt CSS beispielsweise die Einstellung der Zeilenhöhe, eines Einzuges und anderer typo-

grafischer Parameter. Auch die Größe und Platzierung von Elementen lässt sich nahezu beliebig bestimmen.

Stylesheets = Formatvorlagen

Sie können Stylesheets mit den Formatvorlagen von Textverarbeitungsprogrammen wie Microsoft Word vergleichen: Sie weisen logisch markierten Dokumentbestandteilen, etwa einer Überschrift, bestimmte Erscheinungsmerkmale zu, beispielsweise eine Schriftgröße oder -farbe.

1.2.1 Kurze Geschichte der CSS

CSS Level 1

Die erste Version der Cascading Stylesheets wurde 1996 als CSS Level 1 vom World Wide Web Consortium (W3C) verabschiedet. Der Umfang war damals noch recht überschaubar. Die Spezifikation umfasste nur 76 Seiten und beschränkte sich hauptsächlich auf Textgestaltung und Farben. Netscape hatte einen eigenen Vorschlag beim W3C eingereicht – JavaScript Stylesheets – und diesen auch gleich im 1997 erschienenen Netscape Navigator 4 umgesetzt.

1996 erschien auch der Internet Explorer 3, bei dem erstmals eine Art CSS-Unterstützung vorhanden war.

1997 wurde der Wirkungsbereich von CSS auf Layoutaufgaben ausgedehnt – das W3C veröffentlichte ein Arbeitspapier zu CSS-P (CSS positioning). Später wurden die darin beschriebenen Eigenschaften in CSS Level 2 übernommen.

Sie finden alle Spezifikationen von CSS 1 auf der beiliegenden CD-ROM.

CSS Level 2

Die bis heute aktuelle Version der Cascading Stylesheets ist CSS Level 2 – verabschiedet als offizielle Empfehlung des W3C im Mai 1998. Die Spezifikation ist auf knapp 340 Seiten angewachsen und beschreibt neben dem Aussehen von HTML-Dokumenten auch ihre Erscheinung in anderen Medien wie Screenreadern oder auf dem Drucker.

Sie finden alle Spezifikationen von CSS 2 auf der beiliegenden CD-ROM.

CSS 2.1

Inzwischen hat das W3C ein Update für CSS 2 vorgestellt. In CSS 2.1 werden einige Fehler der ursprünglichen Spezifikation behoben und einige Unklarheiten ausgeräumt. Ansonsten passt das W3C die Spezifikation hauptsächlich der Browserrealität an und erklärt einige Eigenschaften, die ohnehin nicht unterstützt wurden, zu optionalen

Spezifikationen. CSS 2.1 hat zum Zeitpunkt der Manuskripterstellung den Status »Candidate Recommendation« – es ist noch nicht offiziell verabschiedet (das wäre dann »W3C Recommendation«), steht aber kurz davor.

Sie finden die zum Zeitpunkt der Manuskripterstellung aktuellen Spezifikationen von CSS 2.1 auf der beiliegenden CD-ROM.

CSS Level 3

In der Bearbeitung befindet sich momentan CSS Level 3, welches die Unterstützung fortgeschrittener Layouttechniken und die Einbindung alternativer Medienformate weiter verbessern soll. Über den aktuellen Stand informiert die Website des W3C unter: **http://www.w3.org/style/**

Stylesheets werden immer wichtiger

In der Weiterentwicklung von HTML zu XML spielen Stylesheets sogar eine noch größere Rolle: Hier kann durch Stil-Anweisungen festgelegt werden, welche Elemente des Dokumentes für das betreffende Endgerät überhaupt angezeigt werden. Eine Adressdatenbank würde bei Ausgabe auf einem Webbrowser beispielsweise Bilder der Personen anzeigen, während die gleiche Datenbank bei Abruf über ein Handy die – für eine Anzeige auf dem winzigen Display viel zu großen – Bilder nicht zeigt.

1.3 Wie sehen CSS aus?

Ein Stylesheet ist eine Sammlung von Anweisungen, die beschreiben, wie bestimmte Elemente eines HTML-Dokumentes ausgegeben werden sollen.

Eine CSS-Anweisung besteht immer aus zwei Teilen:

Einem **Selektor**, der beschreibt, *was* formatiert werden soll, und einer **Deklaration**, die beschreibt, *wie* dies geschieht.

Schema einer CSS-Anweisung

Die Deklaration selbst besteht wiederum aus zwei Teilen: der **Eigenschaft** und dem dieser Eigenschaft zugeordneten **Wert**. Dabei befindet sich der Selektor stets links und rechts die Deklaration in einer geschweiften Klammer.

Mit CSS können Sie sowohl bestehende HTML-Elemente umdefinieren, also ihnen ein neues Erscheinungsbild zuweisen, als auch vollkommen eigene Formatzuweisungen, *Klassen* genannt, erstellen.

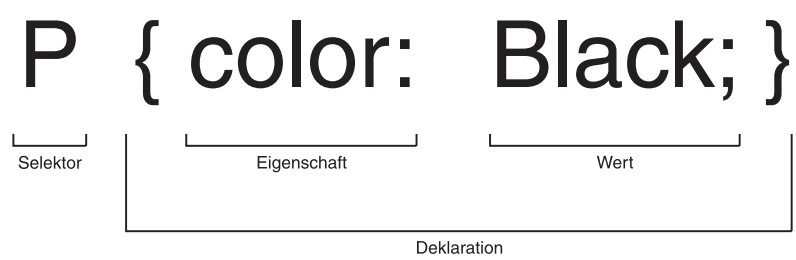

Abbildung 1.8 Schema einer CSS-Anweisung

Einfache CSS-Beispiele

Beispiel: `p { color: black }`

legt als Farbe für den Text des HTML-Elementes `<p>` Schwarz fest.

```
.fussnote { color: #cccccc }
```

erstellt eine neue Klasse mit dem Namen »`fussnote`« und weist ihr die Textfarbe (Vordergrundfarbe) mit dem hexadezimalen Wert *cccccc* (entspricht einem mittleren Grau) zu. Was Klassen sind und wie damit umgegangen wird, zeige ich Ihnen im nächsten Kapitel.

In einem Selektor können mehrere Elemente aufgeführt werden, auf die sich die folgende Deklaration bezieht. Sie werden dann durch Kommata getrennt.

Beispiel: `p, h1 { color: black }`

legt als Farbe für den Text der Elemente `<p>` und `<h1>` Schwarz fest.

In einer Deklaration können gleichfalls mehrere Eigenschaften und Werte zusammengefasst werden.

Beispiel: `p { color: black; background-color: green }`

legt für das Element `<p>` Schwarz als Textfarbe und Grün als Hintergrundfarbe fest. Hierbei ist das Semikolon das Trennzeichen.

Die Anweisungen können entweder direkt in HTML-Tags integriert werden

```
<h1 style="color: black; background-color: green;">Eine
schwarze Überschrift auf grünem Grund</h1>
```

oder gesammelt notiert werden – in einem Stylesheet.

Stylsheets

Stylesheets können im `<head>`-Bereich einer Webseite notiert werden. Die Kommentarzeichen dienen dem Schutz vor alten Browsern, die

kein CSS verstehen und die CSS-Anweisungen sonst einfach anzeigen würden.

```
<head>
   <style type="text/css">
   <!--
      h1 { color: black }
      .fussnote { color: black; background-color: green; }
   -->
   </style>
</head>
   ...
```

Listing 1.1 Einbindung eines Stylesheets im Dateikopf eines HTML-Dokumentes

Stylsheets einbinden

Die Anweisungen können auch in eine Datei ausgelagert und über einen Verweis mit einer oder mehreren HTML-Dateien verknüpft werden.

```
<head>
   <title>Name des Dokumentes</title>
   <link rel="stylesheet" type="text/css"
   href="styles.css">
</head>
...
```

Listing 1.2 Verlinkung eines Stylesheets per Link-Befehl

Eine externe CSS-Datei ist eine einfache Textdatei mit der Endung ».css«.

1.4 »Hallo Welt!« auf CSS

Sehen wir uns eine einfache Webseite einmal im Vergleich an.

Traditionell ist das erste Beispiel für Programmiersprachen die Ausgabe des Textes »Hallo Welt!«. Obwohl CSS und HTML keine Programmiersprachen sind, schließen wir uns da einfach an – ein einfacheres Beispiel ist auch kaum vorstellbar.

Zunächst verwenden wir reines, »logisches« HTML:

Logisches HTML

```
1:    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
2:    Transitional//EN">
3:    <html>
4:       <head>
```

```
5:          <title>Hallo Welt!</title>
6:       </head>
7:       <body>
8:          <h1>Ein einfaches Beispiel</h1>
9:          <p>Hallo Welt!</p>
10:      </body>
11:   </html>
```

Listing 1.3 »Hallo Welt« mit einfachem HTML

Abbildung 1.9 »Hallo Welt« mit HTML-Mitteln

Nun verwenden wir eine Tabelle zum Layouten und den `<font>`-Tag, um die Schriften aufzupeppen: Wir wählen eine andere Schriftart, färben den Seitenhintergrund und fügen Seitenränder hinzu.

HTML als Gestaltungsmittel

```
1:    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
2:    Transitional//EN">
3:    <html>
4:       <head>
5:          <title>Hallo Welt!</title>
6:       </head>
7:       <body bgcolor="#FFCC66" text="#006633">
8:          <table width="100%" border="0" cellspacing="0"
9:          cellpadding="0">
10:            <tr>
11:               <td width="5%"> </td>
12:               <td><b><font color="#990033" size="5"
13:               face="Comic Sans MS">Ein einfaches
14:               Beispiel</font></b></td>
15:            </tr>
16:            <tr>
17:               <td> </td>
18:               <td> </td>
```

```
19:            </tr>
20:            <tr>
21:               <td> </td>
22:               <td>Hallo Welt!</td>
23:            </tr>
24:         </table>
25:      </body>
26:   </html>
```

Listing 1.4 HTML als Gestaltungsmittel

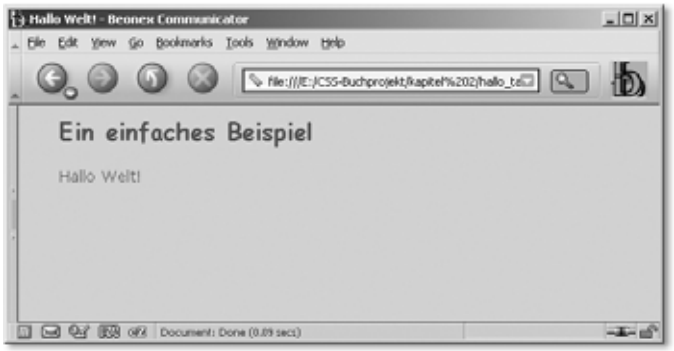

Abbildung 1.10 Mit HTML-»Tricks« gestaltete Webseite

Der Preis für das »anspruchsvollere« Design ist ein um mehr als 100% vergrößerter HTML-Code und damit eine verdoppelte Ladezeit, verlorene Übersichtlichkeit und schlechtere Zugänglichkeit für verschiedene Browserimplementierungen und Geräte sowie die Suchmaschinen.

Verwenden wir dagegen Cascading Stylesheets für die Formatierung, vergrößert sich der HTML-Code gegenüber der ersten Version nur um eine Zeile, die zur besseren Übersichtlichkeit im Folgenden fett ausgezeichnet wird.

Logisches HTML + CSS

```
1:    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
2:    Transitional//EN">
3:    <html>
4:       <head>
5:          <title>Hallo Welt!</title>
6:          <link href="hallo.css" rel="stylesheet"
7:          type="text/css">
8:       </head>
9:       <body>
10:         <h1>Ein einfaches Beispiel</h1>
11:         <p>Hallo Welt! (nur mit CSS)</p>
```

```
12:      </body>
13:   </html>
```

Listing 1.5 HTML für die Struktur und CSS für das Aussehen

Sämtliche Formatierungsinformationen befinden sich in der ausgelagerten Datei *»hallo.css«*, die so aussieht:

```
body {
   background-color: #FFCC66;
}
h1 {
   font-family: "Comic Sans MS";
   font-size: x-large;
   color: #90033;
   margin-bottom: 15px;
   margin-left: 5%;
}
p {
   color: #006633;
   font-size: 0.85em;
   margin-left: 5%;
}
```

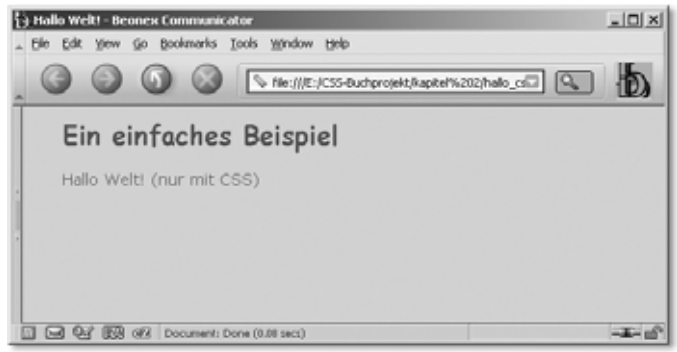

Abbildung 1.11 Gleiches Aussehen bei reduziertem Code mit CSS

Vorteile bei Änderungen

Bei diesem einfachen Beispiel zeigt sich die Einsparung zwar noch nicht, aber denken Sie einmal an eine Website mit mehreren hundert oder gar tausend Dateien, die auf dieselbe Weise formatiert werden. Anstatt in jeder Datei die Formatierung im HTML-Code vorzunehmen, wird dasselbe Stylesheet in allen Dateien einfach eingebunden. Auch Änderungen werden zum Kinderspiel, da nur noch eine einzige Datei geändert werden muss!

CSS mit Dreamweaver

Müssen Sie nun wie in den grauen Vorzeiten von HTML alle CSS-Dokumente wieder per Hand schreiben?

Nein. Moderne Website-Editoren – beispielsweise Dreamweaver von Macromedia – bieten eine mehr oder weniger komfortable CSS-Unterstützung, sodass Sie auch hier nicht auf den (inzwischen) gewohnten Komfort beim Erstellen von Webseiten verzichten müssen.

Trotzdem ist es auch und gerade bei Stylesheets wichtig zu wissen, warum eine bestimmte Anweisung so und nicht anders funktioniert, und beim derzeitigen Stand der Entwicklung ist auch immer wieder das Eingreifen per Hand erforderlich.

Aber sehen wir uns doch einmal unser »Hallo Welt«-Beispiel mit Dreamweaver an.

Sie finden auf der beiliegenden CD-ROM eine Testversion von Dreamweaver MX 2004, der allerneuesten Ausgabe des Editors (leider war zum Zeitpunkt der Manuskripterstellung noch keine Version in deutscher Sprache verfügbar). Für dieses Beispiel installieren Sie Dreamweaver und öffnen die ebenfalls auf der CD-ROM befindliche Datei *hallo_html.htm*.

Sie sollten dann in etwa folgendes Bild (siehe Abbildung 1.12) im Editor sehen (ggf. müssen Sie auf den im Bild umrandeten Knopf drücken).

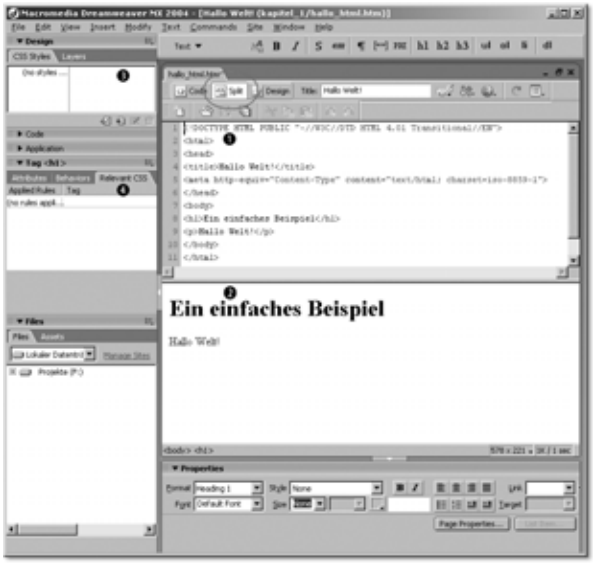

Abbildung 1.12 Hauptfenster von Dreamweaver

Lassen Sie sich nicht von den vielen Optionen irritieren – für uns sind hier vor allem *das Quellcode-Fenster* ❶, die *Layout-Ansicht* ❷ sowie die Reiter *Design* ❸ und *Tags* ❹ wichtig. Eine detailliertere Beschreibung von Dreamweaver MX 2004 finden Sie auf der CD-ROM sowie in Kapitel 8, *Tools für CSS*.

Als Erstes weisen wir dem Dokument die Hintergrundfarbe und die Schriftformatierung zu. Dazu rufen Sie den Menüpunkt **Modify • Page Properties** auf. Hier können Sie (eine Neuerung in Dreamweaver MX 2004) allgemeine Einstellungen für Hintergrund, Schriften und Linkfarben auswählen.

Mit dem Farbpicker wählen Sie hier die Hintergrundfarbe für die Seite aus (siehe Abbildung 1.13).

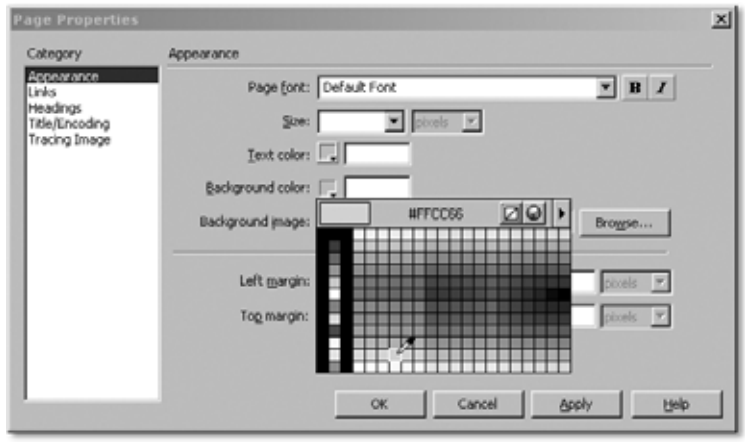

Abbildung 1.13 Auswahl der Seiteneigenschaften per CSS

Das Ergebnis sollte dann wie in Abbildung 1.14 aussehen.

Dreamweaver MX 2004

Im Unterschied zur vorherigen Version nutzt Dreamweaver MX 2004 von sich aus die Formatierung per CSS: Während »damals« nämlich eine allgemeine Hintergrundfarbe in den BODY-Tag geschrieben wurde (`<body bgcolor="#ffcc66" ...>`), sehen Sie hier im Quellcode-Fenster die Hintergrundfarbe als Stil-Anweisung. Im Tag-Reiter finden Sie alle relevanten CSS-Anweisungen, die sich auf den gerade im Editor markierten Tag auswirken ❺.

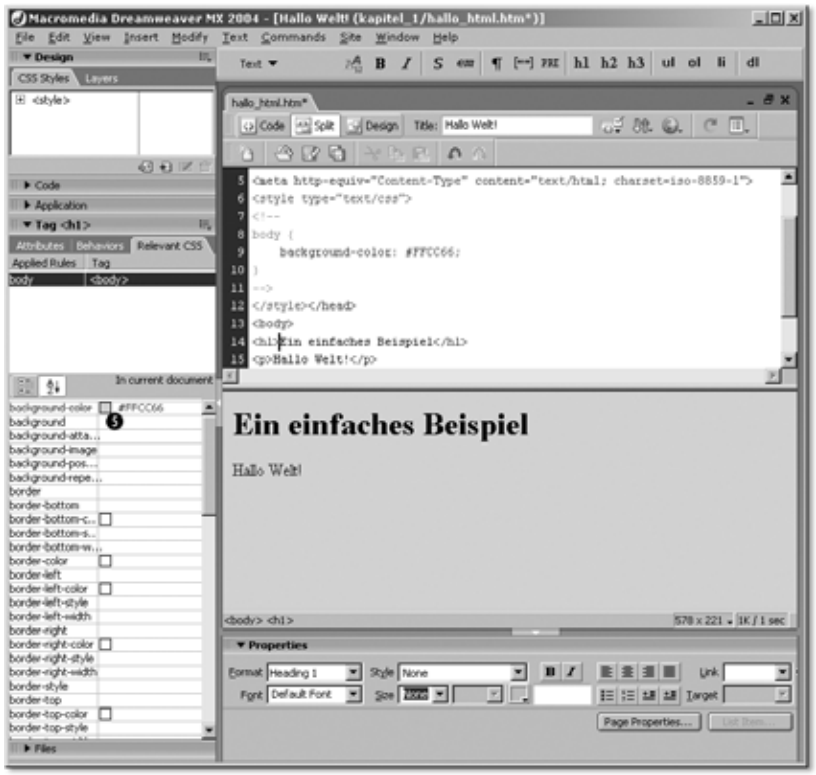

Abbildung 1.14 Hintergrundfarbe per CSS

Als Nächstes formatieren wir die Überschrift mit »Comic Sans MS« in x-large. Dazu klicken Sie im Reiter Design ❸ unten das zweite Symbol von links an (**New CSS Style = Neuen Stil erstellen**).

Zunächst müssen wir uns entscheiden, ob wir eine Klasse erstellen wollen, einen HTML-Tag umdefinieren oder einen »fortgeschrittenen« Selektor (Letzteres ist neu in MX 2004) definieren. Da wir die Überschrift bearbeiten wollen, entscheiden wir uns für den HTML-Tag und wählen in der Dropdown-Liste den Tag `h1` aus.

Unten legen wir noch fest, dass wir ein externes Stylesheet erstellen wollen und wählen (im nächsten Schritt) einen Namen (»*hallo1.css*«). In der nun erscheinenden Dialogbox können wir die Schrift einstellen und die Ränder festlegen (Abschnitt »Box«). Später werden die Ränder noch genauer erläutert – hier wählen wir für `margin-bottom: 15px` und für `margin-left: 5%`. Die anderen Felder lassen wir leer – ein eventuell gesetztes Häkchen bei »same for all« müssen wir vorher entfernen.

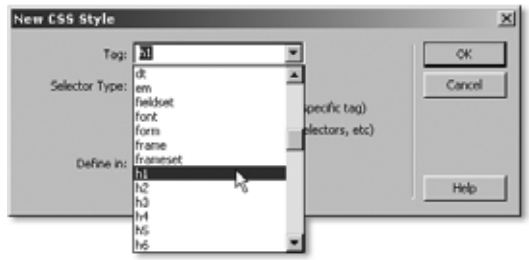

Abbildung 1.15 Neue Formatierung für <h1>

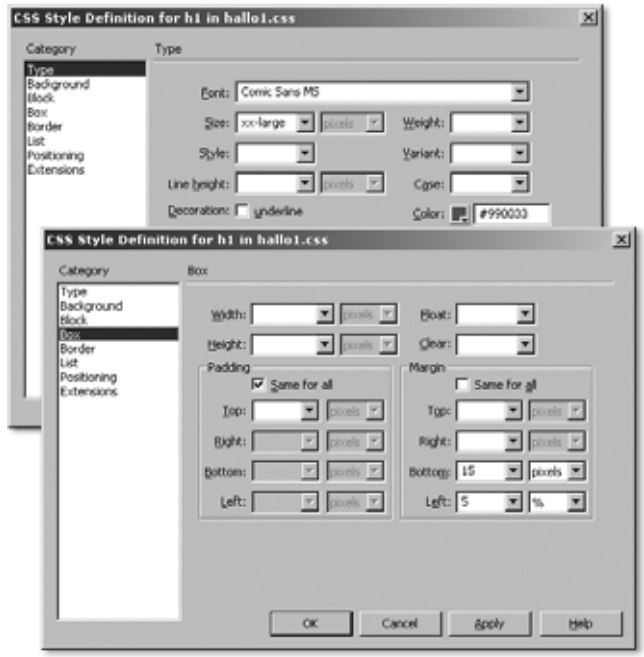

Abbildung 1.16 Schrifteinstellungen und Abstände für »h1«

Unser neu erstelltes Stylesheet wird nun im Design-Reiter aufgeführt. Wenn Sie auf das Kästchen mit dem »+«-Symbol klicken, sehen Sie unseren ersten Stil.

Wir müssen nur noch den Stil für den normalen Text definieren. Dazu klicken Sie im Design-Reiter mit der rechten Maustaste auf den Dateinamen des Stylesheets und wählen im Kontextmenü »Neu« aus. Der Rest ist wie gehabt – die gleichen Dialogfenster erlauben uns, für die Schrift die Farbe `#006633` und eine Schriftgröße von `0.85em` zu wählen. Als `margin-left` stellen wir `5%` ein.

Fertig! Wenn wir die Datei nun speichern (als »*hallo_css2.htm*«) und im Browser öffnen, sieht es genauso aus wie die beiden Vorgänger.

Wenn Sie den Code exakt so haben wollen wie in der Variante »CSS per Hand«, müssen Sie die Anweisungen für `<body>`, die Dreamweaver automatisch in das HTML-Dokument eingefügt hat, dort ausschneiden und in die CSS-Datei einfügen. Dazu öffnen Sie einfach das Stylesheet selbst in Dreamweaver:

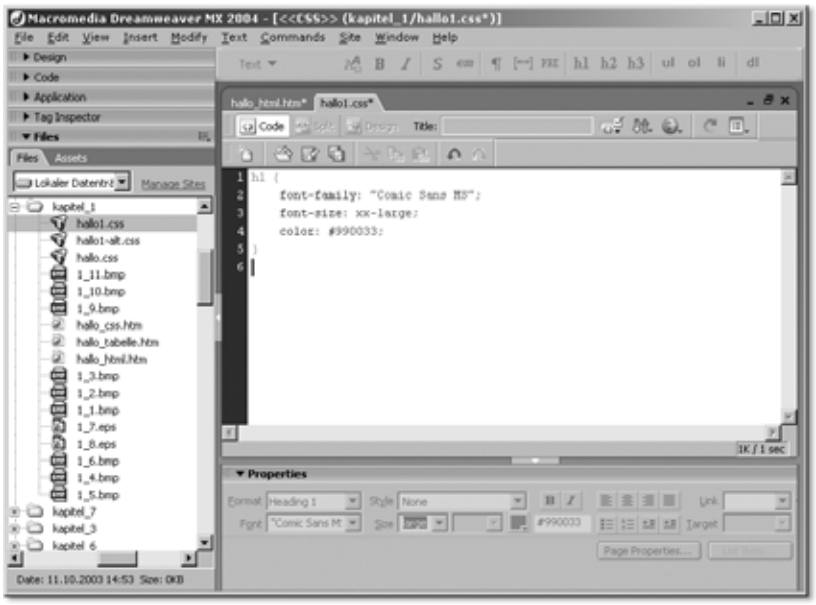

Abbildung 1.17 Stylesheets in Dreamweaver MX 2004 direkt bearbeiten

Sie sehen, Dreamweaver MX 2004 lässt sich auch als CSS-Quellcode-Editor verwenden. Mehr über Dreamweaver und seine CSS-Fähigkeiten erfahren Sie in Kapitel 8, *Tools für CSS*.

1.5 Vorteile und Grenzen des CSS-Einsatzes

Der Einsatz von Cascading Stylesheets bietet eine Reihe von Vorteilen:

- **Einfachere HTML-Dokumente**
  Durch die Trennung von Gestaltung und Inhalt wird eine HTML-Seite kleiner und übersichtlicher. Spätere Änderungen werden einfacher.

- **Mehr Kontrolle**
  Die CSS-Anweisungen erlauben mehr Kontrolle über die Formatierung eines Dokumentes. Gerade typografisch wichtige Anweisungen, wie die Angabe der Zeilenhöhe, sind mit reinem HTML gar nicht möglich.
- **Neue Möglichkeiten**
  Mit CSS ergeben sich neue Gestaltungsmöglichkeiten, die mit reinem HTML nicht zu erreichen sind. Insbesondere das Ein- und Ausblenden von Elementen und die Möglichkeiten fixer Layouts (Kapitel 5, *CSS in der Praxis*) eröffnen neue Horizonte.
- **Höhere Entwicklungsgeschwindigkeit**
  Da auf Tricks wie Layout-Tabellen und »Blinde GIFs« verzichtet werden kann, wird auch die Entwicklung selbst deutlich beschleunigt. Sie wollen einen Kasten mit einem 1 Pixel breiten roten Rahmen? Mit HTML müssen Sie dazu ineinander verschachtelte Tabellen basteln – mit CSS genügt die Anweisung: `border: 1px solid red`
- **Zugängliche Websites**
  Durch den Verzicht auf Formatierungsanweisungen im HTML-Code werden Ihre Seiten auch für Menschen mit Behinderungen oder speziellen Anzeigegeräten (Organizer, WebTV) besser erreichbar. Zugänglichkeit oder Barrierefreiheit (englisch: *accessibility*) wird in den nächsten Jahren auch in Deutschland für Websites immer wichtiger werden. Gesetzliche Bestimmungen, die zumindest für staatliche Websites Barrierefreiheit fordern, sind bereits verabschiedet.
- **Zukunftssicherheit**
  Erfreulicherweise orientieren sich die wichtigen Browserhersteller immer mehr an den Standards für Websites, die vom W3C veröffentlicht werden. CSS ist ein solcher Standard. Websites, die sich an diese Standards halten, müssen nicht befürchten, mit der nächsten Browsergeneration nicht mehr zu funktionieren, wie es beim Einsatz von HTML-Tricks schon einmal passieren kann.
- **Qualitätsbeweis**
  Für professionelle Webentwickler ist auch die Frage der Gewährleistung wichtig. Wenn es einmal zum Streit um die Qualität einer Website kommt, woran lässt sich die Güte von HTML-Code erkennen? Ein Browser, mit dem eine Website schrecklich aussieht, lässt sich immer finden. Da kann es sehr hilfreich sein, internationale Standards bei der Erstellung berücksichtigt zu haben.

Trotz aller Fortschritte gibt es auch für den Einsatz von Stylesheets Grenzen.

Nutzen von CSS noch eingeschränkt

Nicht alle Browser – und erst recht nicht alle anderen Endgeräte – verstehen alle Stylesheet-Angaben. Alte Browser können mit Stylesheet-Angaben gar nichts anfangen oder unterstützen sie nur teilweise. Generell hinken selbst die aktuellen Browser der Entwicklung der Stylesheet-Definition deutlich hinterher. So ist CSS Level 2 längst verabschiedet, wird jedoch selbst von aktuellen Browsern nicht hundertprozentig umgesetzt.

Zusätzlich gibt es Programmfehler (Bugs) in der CSS-Implementierung nahezu aller Browser, die dazu führen, dass syntaktisch korrekte CSS-Anweisungen nicht so ausgeführt werden, wie sie eigentlich laut CSS-Definition angezeigt werden sollten. In den Kapiteln 1, *Einführung in CSS*, und 4, *Browser-Kompatibilität*, werde ich auf solche Probleme detailliert eingehen. Die Verwendung von Grafiken als Text oder Layout-Hilfsmittel (z.B. um Überschriften in bestimmten Schriftarten zu erreichen) schränkt den Nutzen von Stylesheets ebenfalls ein, da das Aussehen und die tatsächliche Größe von Grafiken nicht per Stylesheet verändert werden können – wohl allerdings die Ausrichtung. Es ist auch immer noch nicht ohne weiteres möglich, beliebige Schrifttypen in HTML-Dokumenten zu verwenden.

Und schließlich sind nicht alle Dokumente sinnvoll auf andere Formate übertragbar. So lässt sich eine Tabelle mit zehn Spalten auch mit den besten Stylesheets nicht so formatieren, dass sie auf einem Handy-Display sinnvoll angezeigt werden kann.

5 CSS in der Praxis

5 CSS in der Praxis

5.1 Strategien für den CSS-Designer

Grundsatz für die Arbeit mit Stylesheets ist die Erstellung valider, d.h. syntaktisch korrekter Dokumente – am besten in XHTML.

Valide Dokumente

Eigentlich ist es ja banal; fehlerfreie Dokumente sollten eigentlich eine Selbstverständlichkeit sein. Aber ein Blick in die Quelltexte vieler, auch großer Websites belehrt uns schnell eines Besseren (Schlechteren).

5.1.1 Valide Dokumente erstellen

Durch die rasche Veränderung der für Webdesign wichtigen Sprachen (HTML, XHTML, XML, JavaScript, diverse Skriptsprachen, ActionScript), durch die Vermischung von »offiziellen« Sprachbestandteilen und proprietären Erweiterungen der Browserhersteller, durch die Fehlertoleranz vieler Browser und durch die vielen Bugs, die komplizierte Work-arounds erforderlich machten, hat sich bei vielen Webdesignern die pragmatische Haltung »Richtig ist, was funktioniert« durchgesetzt. Eine weitere Ursache für »schlechten« Code sind auch grafische Editoren wie Frontpage, GoLive und Dreamweaver, die zumindest in den ersten Versionen stark verbesserungsfähigen HTML-Code produzierten (vom MS Word für Windows-HTML-Export gar nicht zu sprechen).

Gründe für standardkonforme Websites

Dabei gibt es genug gute Gründe, den Mehraufwand für syntaktisch korrekte, geordnete Dokumente auf sich zu nehmen:

- **Zukunftssicherheit**
  Glücklicherweise ist die Standardtreue der Browserhersteller deutlich besser geworden. Standardkonforme Webseiten werden daher mit hoher Sicherheit auch in Zukunft von den wichtigen Browsern korrekt angezeigt werden. Nichts ist ärgerlicher, als eine schon fertige Website nachträglich umzubauen, weil ein »schlauer Trick« im neuesten Browser nicht mehr funktioniert (und bei Auftraggebern macht es auch keinen besonders guten Eindruck).
- **Fehlersuche**
  Standardkonforme Dokumente lassen sich automatisch auf Fehler untersuchen. Viele Probleme sind auf einfache Tippfehler zurückzuführen. Der HTML-Validator (Abschnitt 5.1.2) zeigt solche Fehler sofort und erspart die mühselige Suche per Hand.
- **Qualitätssicherung**
  Standardkonforme Dokumente sind auch für Kunden ein Zeichen von Qualität. Besonders in Streitfällen ist es sinnvoll, als Qualitäts-

kriterium für das erstellte Werk einen anerkannten Standard vorweisen zu können.

- **Accessibility**
  Standardkonforme Dokumente sind ein erster Schritt (allerdings auch nur das) hin zu behindertenfreundlichen und zugänglichen Websites. Dies ist nicht nur eine moralische Frage, sondern ist für bestimmte Bereiche jetzt schon gesetzlich gefordert.

Nachteile von standard-gemäßem Arbeiten

Allerdings gibt es im praktischen Einsatz auch Nachteile: Vor allem durch die schlechte Standardunterstützung in der Vergangenheit tummeln sich noch viele Browser auf dem Markt, die solche Dokumente fehlerhaft anzeigen – allen voran der Netscape Navigator 4.x. Im konkreten Einzelfall kann das bedeuten, entweder ein Dokument korrekt standardkonform und valide zu erstellen und dafür eine in älteren Browsern schlechtere Darstellung in Kauf zu nehmen oder Abstriche an der »reinen Lehre« zu machen. Glücklicherweise gibt es auch bei der Standardtreue unterschiedliche »Härtegrade«, wie wir später sehen werden.

Was ist erforderlich für eine valide Webseite?

Dokumententyp angeben

Es ist besonders bei neueren HTML-Sprachversionen wichtig, den richtigen Dokumententyp anzugeben, damit der Browser (oder ein anderes Ausgabegerät) den folgenden Code korrekt interpretieren kann. Ein fehlender oder fehlerhafter Dokumententyp schaltet einige Browser in einen besonderen Anzeigemodus, den *»quirks mode«*, mit dem versucht wird, das Anzeigeverhalten alter Browser nachzuahmen – siehe dazu auch Kapitel 4, *Browser-Kompatibilität*.

5.1.2 Der W3C-Validator

Ein ausgezeichnetes Hilfsmittel, um den korrekten Aufbau eines (X)HTML-Dokumentes zu testen, ist der vom W3C bereitgestellte Validator. Er testet HTML-, XHTML- und XML-Dokumente auf ihre syntaktische Korrektheit und Vollständigkeit.

Es gibt ihn in einer Online-Variante, die sich zum schnellen Testen von Einzeldokumenten eignet, sowie als Download-Variante, die auch bei einigen HTML-Editoren eingebunden ist.

Die Bedienung ist einfach: Sie wählen das zu überprüfenden Dokument durch Eingabe der URL aus (es gibt dort auch eine Maske, in der Sie ein Dokument hochladen können). Encoding und Dokumententyp sollten

in einem validen Dokument angegeben sein – daher können Sie diese Einstellungen auf »automatisch ermitteln« lassen. Nun können Sie noch angeben, ob Sie den Quellcode, eine Zusammenfassung auf Grundlage der logischen Struktur der Überschriften und den Dokumentenbaum anzeigen lassen und ob Sie Attribute mit überprüfen wollen.

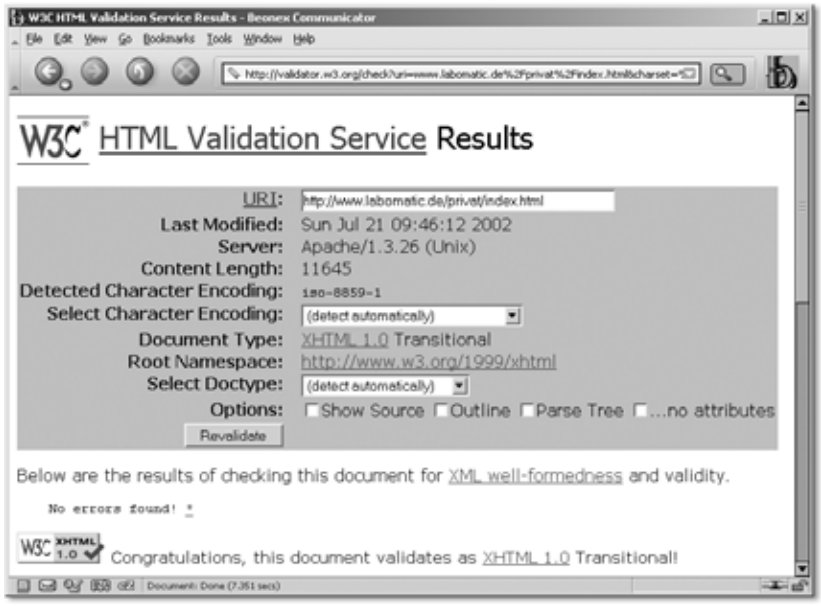

Abbildung 5.1 Gratulation vom W3C – das geprüfte Dokument ist valide!

Dann klicken Sie auf **Validate** und erfahren, ob Ihr Dokument den Segen des W3C erhält.

Sie finden den Online-Validator unter: **http://validator.w3.org**

5.1.3 CSS zur Schriftgestaltung – »Weicher CSS-Einsatz«

Eine relativ unkomplizierte und daher auch für Einsteiger geeignete Methode bei der Nutzung von Stylesheets ist es, erst einmal nur die Eigenschaften zu verwenden, die in nahezu allen Browsern korrekt oder zumindest ohne gravierende Fehldarstellungen angezeigt werden.

Das sind vor allem diejenigen Eigenschaften, die sich auf die Schriftgestaltung beziehen. Zum einen werden diese Eigenschaften in fast allen Browsern größtenteils richtig wiedergegeben, zum anderen fällt es in der Regel weniger auf, wenn eine Anweisung zur Schrift nicht hundertprozentig korrekt angezeigt wird.

:link	
font-family	[Schriftfamilie(n)], serif, sans-serif, monospace
font-style	normal, italic
font-weight	normal, bold
font-size	[absolute Größenangabe], [relative Größenangabe], [Prozentwert], [Schlüsselwörter], larger, smaller
text-decoration	underline, line-through, none
text-transform	capitalize, uppercase, lowercase, none
white-space	normal, pre
line-height	normal
text-indent	[Länge], [Prozentwert]
text-align	left, right, center
color	
background-image	[url], none
background-repeat	repeat, no-repeat
border-width	thin, medium, thick, [Breite]
border-style	solid, double, groove, ridge, inset, outset, none
border	[Farbangabe]
list-style-type	disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha
list-style	[Schlüsselwort]
display	none

Tabelle 5.1 Tabelle der unproblematischen Eigenschaften

Viele der Einschränkungen sind lediglich auf den Netscape Navigator 4 zurückzuführen. Eigenschaften, die hier fehlen, können im Einzelfall durchaus erfolgreich eingesetzt werden – wenn man gewillt ist, herumzuprobieren und umfangreiche Tests durchzuführen.

5.1.4 Komplettlayout mit CSS – »Harter CSS-Einsatz«

Deutlich komplizierter ist es, das gesamte Layout einer Website durch CSS zu realisieren. Die Stile-Eigenschaften, die für die Positionierung von Elementen notwendig sind, werden von den verschiedenen Browsern unterschiedlich, fehlerhaft und unvollständig interpretiert. Eine über verschiedene Betriebssysteme und Browser hinweg einigermaßen

gleiche Darstellung lässt sich nur durch browserspezifisch angepasste Stylesheets, diverse Tricks, eingehende Betrachtung der Fehler der verschiedenen Browser und durch sehr viele Tests erreichen.

Wenn Sie einen relevanten Anteil von Netscape 4.x-Nutzern für Ihre Site erwarten, sollten Sie ein komplexeres CSS-basiertes Layout nicht in Erwägung ziehen.

Die folgende Seite besteht vollständig aus per CSS positionierten `<div>`-Bereichen. Äußerlich unterscheidet sie sich nicht von den bekannten Tabellen-Layouts:

Abbildung 5.2 Vollständig auf CSS basierendes Layout

Erst der Blick hinter die Kulissen – sprich: auf den Quellcode – zeigt die Unterschiede (der Quellcode ist aus Gründen der Übersicht leicht gekürzt):

```
1:   <?xml version: "1.0" encoding="iso-8859-1"?>
2:   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
     Transitional//EN" "http://www.w3.org/TR/2000/
     REC-xhtml1-20000126/DTD/xhtml1-transitional.dtd">
3:   <html xmlns="http://www.w3.org/1999/xhtml" lang="de">
```

Der Dokumententyp zeigt: Es handelt sich um ein XHTML-Dokument.

```
4:    <head>
5:       <title>Newsletters -- South African Tourism</title>
6:       <meta http-equiv="content-style-type"
9:       content="text/css" />
10:      <script language="javascript" type="text/javascript"
         src="js/nn_resize.js"></script>
11:      <link rel="stylesheet" type="text/javascript"
         media="screen" href="css/screennn.js" />
12:      <link rel="stylesheet" type="text/css" media="screen"
         ref="css/screennn.css" />
13:      <style type="text/css" media="screen">
14:      @import url(css/screen.css);
15:      </style>
```

Hier eine Reihe von eingebundenen Skripten. Die meisten dienen der Einbindung unterschiedlicher Stylesheets unter besonderer Berücksichtigung des Netscape Navigators 4.

```
17:   </head>
18:   <body>
19:      <div id="head">
20:         <div id="flagSymbol">
            <img src="img/flagsymbol.gif"
            alt="" width="199" height="97" border="0" />
            </div>
21:         <div id="headImage"><img src="img/header.jpg"
            alt="" width="549" height="97" border="0" /></div>
            <div class="nav" id="headFirstRow"><img src="img/
            south_african_tourism.gif" alt="" width="193"
            height="10" border="0" />  <a
            class="active"
            href="index.html" title="Newsletters Home">
            NEWSLETTER</a></div>
22:         <div class="nav" id="headSecondRow"> <a
            href="news_from_germany/index.shtml" title="News
            From Germany Newsletter">NEWS FROM GERMANY</a> | <a
            href="presse-info/index.shtml" title="S&uuml;
            dafrika Presse-Info Newsletter">S&Uuml;DAFRIKA
            PRESSE-INFO</a>
```

```
          | <a href="product_news/index.shtml" title="Product
          News Newsletter">PRODUCT NEWS</a></div>
27:     </div>
```

Der Kasten mit dem Seitenkopf

Der Quellcode ist sehr verständlich benannt, sodass Sie leicht erkennen, welche Bereiche welche Inhalte enthalten.

```
28:     <div class="pageBody" id="pageBodyTwoRowNav">
29:       <div class="content" id="side">
30:       <h3>Newsletters</h3>
31:       </div>
32:       <div class="content" id="main">
33:       <h4 class="top"><a href="news_from_germany/
          index.shtml" title="News From Germany Newslet-
          ter">NEWS FROM GERMANY</a></h4>
35:       <p>News from the German travel market -
          bimonthly.</p>
42:       <p> </p>
43:       <p><a href="newsletter.html">Newsletter abonnie-
44:       ren</a> - <a href="newsletter_e.html">subscribe
          newsletter</a>
          </p>
46:     </div>
47:     <br clear="all" />
```

Seitenkörper mit zwei Spalten

Links befindet sich ein Bereich mit der `id="side"`, der die Überschriften enthält – rechts daneben der Fließtext.

```
48:     <div id="foot">
49:       <div class="nav" id="footFirstRow">  <a
          class="active" href="index.html" title="Newsletters
          Home">NEWSLETTER</a></div>
          <div class="nav" id="footSecondRow"><a
          href="news_from_germany/index.shtml" title="News
          From Germany Newsletter"> NEWS FROM GERMANY</a> |
          <a href="presse-info/index.shtml" title="S&uuml;daf-
          rika Presse-Info Newsletter">S&Uuml;DAFRIKA PRESSE-
```

```
            INFO</a>
            | <a href="product_news/index.shtml" title="Product
              News Newsletter">PRODUCT NEWS</a></div>
54:           <div class="address" id="footThirdRow">
                South African Tourism  
              |  An der Hauptwache 11  |
                60313 Frankfurt</div>
56:           <div class="address" id="footFourth-
              Row">  T:
              0949-69-92 91 29 0  |  F: 0949-
              69-28 09 50  |  <a href="mailto:
              info@southafricantourism.de">
              info@southafricantourism.de</a></div>
58:       </div>
```

Die Fußzeile mit unterem Menü:

```
59:   </div>
60:   </body>
61:   </html>
```

Listing 5.1 Komplettlayout per CSS

Abbildung 5.3 Die CSS-Bereiche markiert

Beim Durchlesen des Quelltextes werden der Aufbau der Seite und die Funktion der jeweiligen Abschnitte sofort offenbar. Stellen Sie sich eine solche Seite einmal als HTML-Tabellenkonstruktion vor. Nicht nur würde der Code deutlich umfangreicher und komplizierter sein, aus seiner Schreibweise ergäben sich auch keinerlei Hinweise auf die Struktur des Dokumentes.

CSS Zen Garden

Welche Möglichkeiten eine konsequente Trennung von Inhalten und Design bietet, demonstriert eindrucksvoll das Projekt CSS Zen Garden: Dort hat Dave Shea eine HTML-Seite bereitgestellt, die vollständig per CSS gestaltet wird, und hat andere Entwickler aufgefordert, ihre CSS-Gestaltungsvorschläge beizusteuern. Inzwischen sind es über 100 und die Bandbreite der Designs ist atemberaubend! Und als Grundlage dient immer dieselbe HTML-Datei!

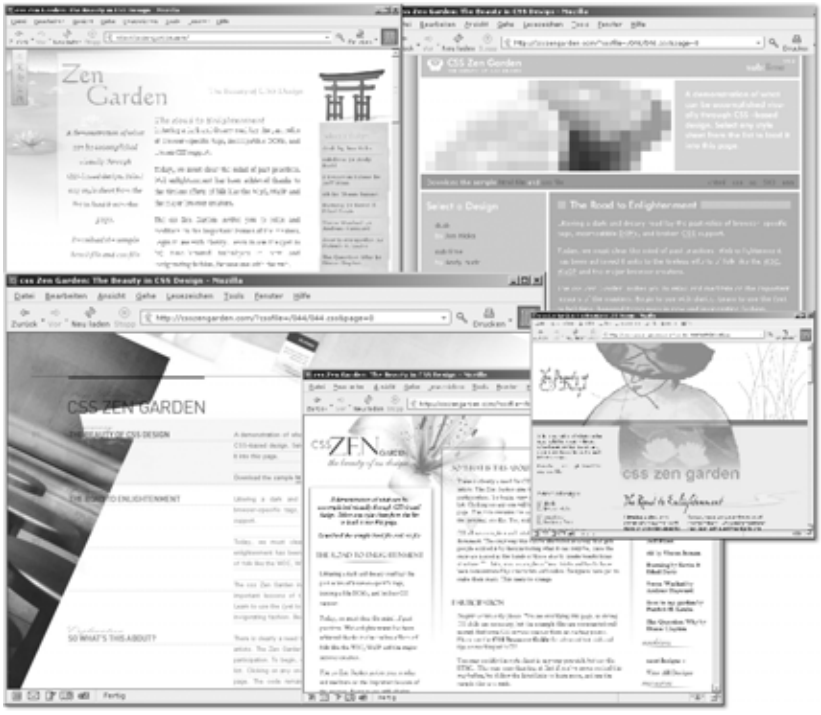

Abbildung 5.4 Die Vielfalt von CSS entdecken – im Zen Garden

http://www.csszengarden.com

5.1.5 Suchmaschinenoptimierung per CSS

Auch für die Positionierung in Suchmaschinen ist der Einsatz von Stylesheets hilfreich: Zum einen verbessert er den Anteil inhaltlichen Textes des Dokumentes (*»content-to-code-ratio«*), zum anderen kann er helfen, grafische Texte Suchmaschinen-tauglich darzustellen.

Inhaltsanteil verbessern

Wie ganz am Anfang des Buches diskutiert, lässt sich durch den Einsatz von CSS einiges an Codezeilen sparen. Insbesondere dann, wenn Gestaltungsanweisungen konsequent in Stylesheets ausgelagert werden, lässt sich der HTML-Code entschlacken. Für die Anzeige im Browser müssen diese Anweisungen natürlich trotzdem geladen werden, aber die Spider der Suchmaschinen ignorieren Stylesheets in der Regel und bekommen von diesen Zeilen nichts mit.

Da Suchmaschinen für die Wertigkeit einer Seite auch den Anteil des Suchwortes am gesamten Text einer Seite berücksichtigen, ist es sinnvoll, den Code einer Seite so schlank wie möglich zu halten. Ganz abgesehen davon ist das natürlich auch gut für eine kurze Ladezeit.

Anstatt `<strong><font face="Arial, Helvetica, sans serif" color="#3333dd" size="3">Dies ist eine besonders wichtige Überschrift</font></strong><br><br>` zu schreiben (143 Zeichen, davon 44 Inhalt = 30%), schreiben wir nur noch `<h1>Dies ist eine besonders wichtige Überschrift</h1>` in das HTML-Dokument (53 Zeichen, davon 44 Inhalt = 83%) und den Rest in ein Stylesheet.

Würde über eine Suchmaschine nach dem Suchwort »Überschrift« gesucht, hätte dieses im ersten Beispiel knapp 8% Anteil am Seiteninhalt. Im zweiten Fall wäre der Anteil mit 20% fast dreimal so groß.

Es lässt sich auch noch einiges mehr an Code sparen, wenn beispielsweise Grafiken nicht mehr mittels `<img>`-Tag eingebunden werden, sondern per CSS. Das bietet sich aus semantischer Sicht allerdings nicht für Grafiken an, die zum Inhalt gehören. Auch hier ein Beispiel:

Anstatt ein Menü mit Aufzählungspunkten so zu realisieren:

```
<a href="link1.html"><img src="icon.gif" height="5"
width="8" alt=""> <font face="Arial, Helvetica, sans
serif" color="#3333dd" size="3">Link1</font></a><br>
<a href="link22.html"><img src="icon.gif" height="5"
width="8" alt=""> <font face="Arial, Helvetica, sans
```

```
serif" color="#3333dd" size="3">Link2</font></a><br>
<a href="link3.html"><img src="icon.gif" height="5"
width="8" alt=""> <font face="Arial, Helvetica, sans
serif" color="#3333dd" size="3">Link3</font></a><br><br>
```

muss nur Folgendes notiert werden:

```
<ul>
   <li><a href="link1.html">Link 1</a></li>
   <li><a href="link2.html">Link 2</a></li>
   <li><a href="link2.html">Link 3</a></li>
</ul>
```

Das ist nicht nur semantisch korrekter und übersichtlicher, sondern auch mit 123 zu 455 Zeichen deutlich kompakter. Im Stylesheet stünde dann z. B.:

```
Li {
   list-style-image: icon.gif;
   font-familiy: Arial, Helvetica, sans serif;
   font-size: 1.5em;
   color: #3333dd;
}
```

Noch dramatischer werden die Unterschiede, wenn wir auf verschachtelte Tabellen für das Layout zurückgreifen ...

Kombinieren lässt sich dies noch mit der folgenden Technik, die eigentlich zur Verbesserung der Zugänglichkeit gedacht war (aber da leider nicht so gut funktioniert wie geplant).

Logische Auszeichnungen statt Grafiken

Suchmaschinen mögen es besonders, wenn ein Dokument die von HTML vorgesehenen logischen Kennzeichnungen verwendet – wenn also ein Suchbegriff in einer Überschrift erster Ordnung ganz am Anfang der Seite steht:

```
<body>
   <h1>Unser Super-Projekt</h1>
   ...
</body>
```

Nun haben wir aber vielleicht für unser »Super-Projekt« auch ein beeindruckendes Logo, das wir unseren menschlichen Besuchern nicht vorenthalten wollen. Normalerweise würden wir dieses an Stelle der H1-Überschrift als Grafik einbinden und den Titel im `alt`-Attribut unterbringen.

```
<body>
   <img src="logo.gif" width="350" height="80" alt=" Unser
   Super-Projekt">
   ...
</body>
```

Sowohl die Suchmaschine als auch Besucher beeindrucken wir mit dieser Variante:

```
<body>
   <h1>Unser Super-Projekt</h1>
   ...
</body>
```

und diesen Stilen:

```
body {
   background-image: url("logo.gif");
   background-repeat: no-repeat;
}
body h1 {
   display: none;
}
```

So erhalten wir im HTML-Dokument einen absolut einfachen und sauberen logischen Code und sehen auf der Website die ganze grafische Pracht. Im realen Einsatz könnten Sie die Grafik dann noch positionieren und ggf. mit einem `margin-top` für `H1` Platz für das Logo schaffen. Einziger Nachteil: Sie können die Grafik nicht mehr als Link zur Homepage verwenden, wie auf vielen Websites üblich.

Eigentlich wäre das auch ein gute Methode für Screenreader und ähnliche Geräte. Leider hat sich gezeigt, dass der weitverbreitete Screenreader JAWS die mit `display: none` ausgezeichneten Elemente auch nicht vorliest ... Die Accessibility-Experten diskutieren noch, wie auf dieses ärgerliche Verhalten am besten zu reagieren ist.

Generell lässt sich dieses Verfahren auf verschiedene Elemente anwenden: Schmuckgrafiken als CSS-Hintergrund, Menüs als Listen, Rollover-Effekte mit `:hover` statt JavaScript und natürlich Positionierung mit CSS statt Tabellen.

5.2 Schriftgrößen

Ein Problem, mit dem sich Webdesigner herumschlagen, seit Netscape mit der Erfindung des `<font>`-Tags Angaben zur Schriftgestaltung ermöglicht hat, ist die Gestaltung von Text. Einerseits stehen dem Designer nur die – relativ wenigen – Schriften zur Verfügung, die der Betrachter einer Webseite auf seinem Rechner installiert hat, andererseits wird die Größe der Schrift auf verschiedenen Browsern und Betriebssystemen unterschiedlich angezeigt. Das liegt daran, dass die Größe von Schriften in der Regel in Punkten (Points) angegeben wird. Dabei hat ein Punkt eine Größe von 1/72 Zoll. Macintosh-Computer verwendeten diese Festlegung auch für die Umrechnung von Schriftgrößen für die Bildschirmdarstellung (72 ppi). Microsoft setzte hingegen eine Auflösung von 96-120 ppi (je nach Einstellung der Schriftgrößen unter den Bildschirmeinstellungen) ein. Das führte dazu, dass Schriftgrößen, die auf PCs noch lesbar waren, unter Macintosh-Browsern nicht mehr zu erkennen waren.

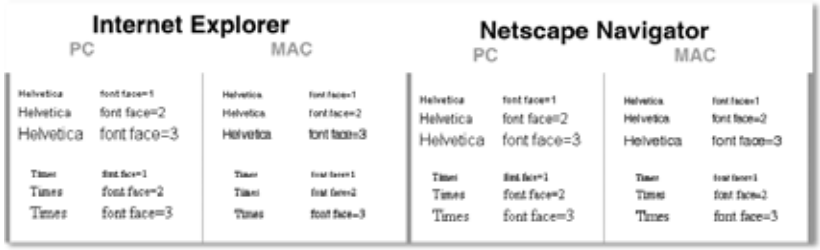

Abbildung 5.5 Unterschiedliche Schriftdarstellung zwischen Browsern und Systemen

Screenshots mit freundlicher Genehmigung von Michael Kaspar (**http://www.upuauet.com**).

Mit der Problematik haben sich auch Rene Grassegger auf seiner WIN-MAC-Site (**http://grassegger.at/winmac/index.htm**) und Fabrice Pascal (**http://www.fabrice-pascal.de/artikel/sizediscussion/**) beschäftigt. Und wem die dort angebotenen Informationen noch nicht reichen, der kann sich die 264(!) Screenshots von Owen Briggs ansehen (**http://www.thenoodleincident.com/tutorials/box_lesson/font/index.html**).

Glücklicherweise hat Apple bei neueren Macintosh-Rechnern (ab Internet Explorer 5.0) die Schriftdarstellung der des PCs angepasst, sodass Schriften dann gleich groß dargestellt werden.

Nachdem der <font>-Tag seit HTML 4 als veraltet gilt und nicht mehr benutzt werden soll (in XHTML existiert er nicht mehr), muss die Formatierung von Schriften per CSS erfolgen.

Schriftgestaltung ohne <font> – Chance statt Einschränkung

Das ist aber eigentlich keine Einschränkung, sondern eine Verbesserung, da in CSS wesentlich mehr Möglichkeiten zur Verfügung stehen, die Typografie eines Textes zu beeinflussen. So lassen sich per CSS die Zeilenhöhe und weitere typografische Merkmale verändern (siehe auch Abschnitt 9.16, *font-family*, bis 9.36, *unicode-bidi* (CSS2)).

Zur Kennzeichnung der Schriftgröße stehen dabei die Eigenschaft `font-size` (Abschnitt 9.20, *font-size*) und vier Arten von Einheiten zur Verfügung:

- Schlüsselwörter (xx-small bis xx-large)
- Relative Angaben (Prozentangaben, em, ex) und
- Punkte oder
- Pixel

Da alle Varianten Vor- und Nachteile aufweisen, haben sich unter den Webdesignern unterschiedliche »Schulen« gebildet, die jeweils eine der Methoden befürworten. Daneben gibt es auch noch Usability-Experten, die generell Angaben zur Schriftgröße ablehnen und die alleinige Kontrolle über die Anzeige von Schriften dem User überlassen wollen. Da die Standardschriften sehr unterschiedlich und meist ziemlich groß angezeigt werden, eignet sich diese Variante tatsächlich nur für sehr einfach gestaltete Seiten. Eine einigermaßen exaktes Layout lässt sich damit kaum erreichen.

5.2.1 Schlüsselwörter

Es gibt in der CSS-Definition sieben absolute Schlüsselwörter:

small x-small xx-small	medium	large x-large xx-large

medium ist die Basisgröße – die anderen Werte werden jeweils um den Faktor 1,2 vergrößert oder verkleinert, xx-small ist also etwa 0,58 mal so groß wie die Basisschriftart (1 / (1,2 x 1,2 x 1,2)).

Unglücklicherweise entspricht die Einstellung *medium* in verschiedenen Browsern nicht der gleichen Schriftgröße, was damit zusammenhängt, dass Microsoft und Netscape die Entsprechung zwischen den

CSS-Schriftgrößen und den früher verwendeten HTML-Schriftgrößen teilweise unterschiedlich handhaben: Während der Internet Explorer 5 die kleinste HTML-Schriftgröße `font size="1"` als `xx-small` einsetzt und dann `medium` mit `font size="4"` gleichsetzt, definiert Netscape `medium` so, dass es der früheren Standardeinstellung `font size="3"` entspricht.

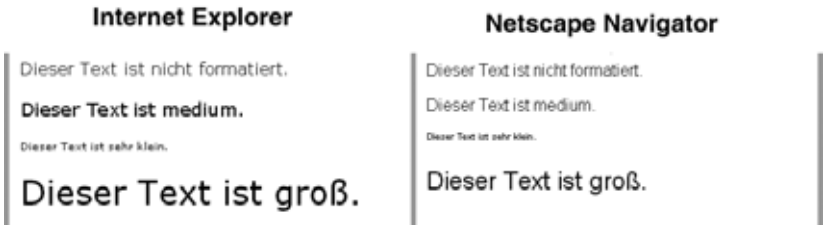

Abbildung 5.6 Unterschiedliche Schriftdarstellung bei der Verwendung von Schlüsselwörtern

Hier hilft nur ein Trick weiter: Mit Tanteks Hack oder den Conditional Comments von Microsoft können Sie allen Browsern »ihre« passenden Schriftangaben zur Verfügung stellen.

Da die normale Einstellung *medium* meist etwas zu groß erscheint, eignet sich *small* recht gut als Standardeinstellung für normalen Text. *small* entspricht bei den meisten Browsern etwa 12px – nur der Internet Explorer 5 benötigt *x-small*, um diese Schriftgröße anzuzeigen.

Kein Problem mit Tanteks Hack:

```
<style type="text/css">
/*<![CDATA[*/
   p {
      font-size: x-small;
      voice-family: "\"}\"";
      voice-family: inherit;
      font-size: small;
      }
   html>p {
      font-size: small;
      }
/*]]>*/
</style>
```

Mit dem aus Abschnitt 4.4.2 bekannten Trick setzen wir dem Internet Explorer 5.x (Win) zunächst »seinen« Wert vor und lassen ihn dann die Interpretation abbrechen. Anschließend bekommen alle anderen Browser den korrekten Wert zugewiesen, der nach den Regeln der Wertigkeit die früheren Angaben überschreibt. Das geht natürlich auch mit Conditional Comments (siehe Abschnitt 4.4.9).

5.2.2 Relative Angaben

Relative Angaben zur Schriftgröße beziehen die Schriftgröße eines Elementes auf sein Eltern-Element und sind daher eigentlich sehr gut geeignet, um die Größenverhältnisse zwischen unterschiedlichen Textelementen zu wahren, dem Benutzer aber die Möglichkeit zu lassen, die Schriftgröße seinen Wünschen anzupassen.

Allerdings machen sie das Erstellen von Stylesheets auch etwas komplizierter, da durch sie die Struktur eines HTML-Dokumentes Einfluss auf die Schriftgröße erhält:

```
<style type="text/css">
   /*<![CDATA[*/
      body { font-size: 0.8em; }
      div { font-size: 0.8em; }
      p { font-size: 0.8em; }
   /*]]>*/
</style>
```

führt dazu, dass die Schriftgröße für alle Elemente auf 0.8em gesetzt wird – ein Absatz hat also eine Schriftgröße von 0,8 x 0,8 = 0,64. Wenn Sie später noch ein `<div>` um den Absatz legen, verkleinert sich die Schriftgröße nochmals auf 0,51em.

Auch Mehrfachnotationen wie

```
body,p,h1,h2,h3,h4,table,
th,td,tr,ul,ol,li { font-family:Arial,Helvetica,
sans-serif; font-size: .8em}
```

– verwendet, weil Netscape 4.x dazu neigt, eigentlich vererbte CSS-Eigenschaften nach bestimmten Elementen zu »vergessen« – führen zu bis ins Unleserliche verkleinerten Schriften.

Ein weiterer Browserbug betrifft den Internet Explorer, der unter bestimmten Bedingungen Schriften unleserlich klein darstellt (lesen Sie dazu das Fazit und Kapitel 7, *Beispiele*).

Trotzdem ist die Auszeichnung mittels relativer Einheiten die sauberste Lösung für flexible und barrierefreie Dokumente.

5.2.3 Punkte

Da Punkte (Points) die aus dem Printbereich bekannte Art sind, Schriftgrößen anzugeben, liegt zunächst einmal der Gedanke nahe, sie auch für Stylesheets zu verwenden. Doch Bildschirme sind eben keine Druckerpressen, und die Verwendung von Punkten führt dazu, dass Schriften auf verschiedenen Betriebssystemen unterschiedlich dargestellt werden und des Öfteren auf Macintosh und Linux-Systemen unleserlich werden. Grund dafür ist wieder der oben schon erwähnte unterschiedliche Umrechnungsfaktor (96/72 ppi).

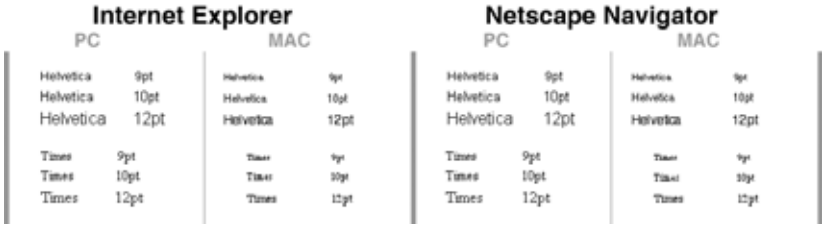

Abbildung 5.7 Vergleich der Point-Darstellung

Schriften unter 9 Punkt sind generell auf älteren Macintosh-Rechnern nicht zu lesen – gerade die recht beliebte Variante, als Schrift Arial in 8 Punkt Größe einzusetzen, führt zu unleserlichem Text für manche Mac-User (zumal viele Macintosh-Rechner gar keine Arial installiert haben und der Text in Helvetica angezeigt wird, die am Bildschirm ohnehin in kleinen Größen schlecht lesbar ist). Glücklicherweise lassen sich die Schriften auf neueren Systemen der Windows-Darstellung angleichen, sodass dieses Problem mit der Zeit verschwinden wird.

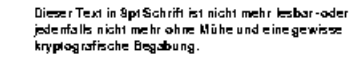

Abbildung 5.8 Schrift bei älteren Macintosh-Rechnern – können Sie etwas erkennen?

Trotzdem sind Punkte nicht die ideale Größe, um Schriften für Websites zu definieren. Anders sieht es natürlich aus, wenn Sie ein Stylesheet für die Druckversion einer Seite (siehe auch Abschnitt 5.9, *Druckversion per CSS)* erstellen. Hier sind Punkte genau richtig – die Seite soll ja gedruckt werden.

5.2.4 Pixel

Letztlich wird alles, was am Bildschirm sichtbar ist, in Pixeln angezeigt. So ist die Einheit Pixel als »natürliche« Einheit für Screendesign die einzige, die (so gut wie) unter allen Systemen und Browsern gleich angezeigt wird. Ein Buchstabe von 12 Pixel Höhe nimmt auf einem Bildschirm mit 800 x 600 Pixel eben immer 2% der Bildschirmhöhe ein – egal welcher Browser ihn darstellt. So schwören viele Designer, die komplizierte, grafiklastige Layouts erstellen, auf Schriftgrößen, die in Pixel definiert sind.

Die Definition in Pixel hat aber einen entscheidenden Nachteil: Im Internet Explorer ist es nicht möglich, in Pixel definierte Schriften zu skalieren. Nur dadurch können aber sehbehinderte Menschen eine für sie angenehme Schriftgröße einstellen.

Andererseits ist es in bestimmten Situationen eben auch sinnvoll, die Schriftgröße unveränderbar zu halten: Wenn Schrift in Menüs verwendet wird, die nur eine bestimmte Breite aufweisen, würde ein Vergrößern der Schrift das Layout sprengen.

Texte, die als Bildunterschriften kleiner Grafiken dienen, sollen in einem passenden Verhältnis angezeigt werden.

5.2.5 Gemischte Angaben

Prinzipiell ist es natürlich auch möglich, verschiedene Einheiten zu mischen. Was bei gleichartigen Elementen kaum sinnvoll ist, da die Verhältnisse zwischen den Schriften sich beim Skalieren verändern.

```
p {font-size: 12px; }
h1 {font-size: 16px; }
h2 { font-size: 1.2em; }
```

Ist im Browser eine Basisschriftgröße von 12px eingestellt, so ist die Überschrift `<h2>` mit 1,2 x 12px = 14,4px kleiner als `<h1>`, was auch ihrer Bedeutung entspricht. Wird aber eine Basisschriftgröße von 16px verwendet, so überragt die `<h2>`-Überschrift mit 19px die `<h1>`. Das ändert den Gesamteindruck des Dokumentes und entspricht auch nicht der durch die Formatierung gesetzten Hierarchie (`<h1>` ist wichtiger als `<h2>` und das sollte auch optisch deutlich sein).

Es gibt jedoch durchaus Situationen, in denen gemischte Angaben sinnvoll sind. In einem ansonsten in Pixel definierten Stylesheet kann eine Angabe

```
.unwichtig {font-size: 80%; }
```

Arbeit und Flüchtigkeitsfehler ersparen, da bei einer Änderung der generellen Schriftgröße diese Deklaration immer noch passt. Außerdem ist sie auf unterschiedliche Elemente anwendbar: ob im 12px großen Fließtext, der 16px großen Überschrift oder einem Kasten, der eine Schriftgröße von 14px besitzt – die Auszeichnung `class="unwichtig"` führt immer zu passend verkleinertem Text.

Sie müssen natürlich im Auge behalten, dass durch die Definition einer kleinen Schrift für ein Element nicht an anderer Stelle Schriftgrößen entstehen, die unterhalb der Lesbarkeitsschwelle liegen.

```
p {font size: 10px;}
```

würde bei einer HTML-Konstruktion von

```
<p>Dies ist ein Text mit <span class="unwichtig">unwichti-
gem Inhalt</span>.</p>
```

den unwichtigen Text in einer Schriftgröße von 10px x 80% = 8px darstellen, die ihn womöglich nicht nur unwichtig, sondern auch unlesbar macht.

Umgekehrt können Sie Schrift für Menübereiche oder Bildunterschriften, die aufgrund eines festen Layout-Bereiches eine bestimmte Breite einnehmen müssen, auch in Pixel angeben, während der Fließtext relative Größenangaben hat. Wenn diese relativ definierten Schriften vergrößert werden, ändert sich natürlich das Verhältnis der Schriften auf der Webseite untereinander, wie oben schon mit den Überschriftenebenen beschrieben. Das ist unter Umständen aber eher akzeptabel als eine Zerstörung des Layouts. Wenn wir davon ausgehen, dass am ehesten sehbehinderte Menschen die Schriftdarstellung deutlich vergrößern, ist es recht sinnvoll, am unteren Rand einer Seite eine alternative Textnavigation anzubieten – dann natürlich auch mit relativen Schriftgrößen.

Was also ist die beste Variante für die Angabe von Schriftgrößen? Es kommt darauf an, welche Art von Website Sie planen und wie der (X)HTML-Code der Seiten aufgebaut ist.

Wann immer möglich: flexible Schriftgrößen

Für eine Site mit variablem flexiblen Layout sind Schlüsselwörter, relative Größen wie em oder Prozentangaben die richtige Wahl. Schlüsselwörter sind recht einfach anzuwenden und unproblematisch in Bezug auf die Dokumentenstruktur. Relative Angaben hingegen erlauben eine feinere Abstufung, erfordern aber auch etwas mehr Aufmerksamkeit bei verschachtelten Dokumenten. Mit einem Skript, wie in Abschnitt 6.41 vorgestellt, sichern Sie sich gegen den »Minischrift-Bug« im Internet Explorer ab.

Inzwischen hat sich im Zuge einer Diskussion über diesen Bug eine Lösung ergeben, die auch ohne komplizierte JavaScripte auskommt.

Die einfache Angabe:

```
body {
  font-size: 101%;
}
```

verhindert den Bug. Hiervon ausgehend können Sie nun problemlos alle weiteren Schriften in Prozent oder em angeben.

Warum 101%? Ein weiterer Bug in Opera 6 führt bei 100% zu Anzeigefehlern – bei 101% funktioniert die Anzeige browserübergreifend.

Gleiche relative Schriften für alle

Nach »Einfach für Alle« werden auch die Schriftgrößenangaben 69%, 76%, 86%, 93% browser- und plattformübergreifend gleich dargestellt und eignen sich daher als Standardschriftgrößen.

Pixelmaße für fixe Layouts

Für Seiten mit pixelexakten Layouts, die mit absolut positionierten Bereichen arbeiten, sind Angaben in Pixel manchmal die bessere Wahl.

Sie sollten Ihren Besuchern dann die Möglichkeit geben, die Schriftgröße zu verändern. Auch hier hilft das Skript aus Abschnitt 6.41. Denken Sie auch daran, dass serifenlose Schriften unter 8px Höhe nicht mehr lesbar sind, eher noch Schriften mit Serifen (wie Times New Roman).

5.3 Positionierung

Webseiten vollständig mit CSS zu layouten ist die »hohe Schule« der CSS-Anwendung. Weniger, weil die Positionierung selbst so komplex ist, sondern eher, weil die verschiedenen Browser einem das Leben durch eine Vielzahl von Bugs und Schwächen schwer machen.

Index

N

O

P

Q

R