

Kai Laborenz



CSS-Praxis



Auf einen Blick

Vorwort	13
1 Einführung in CSS	17
2 Grundlegende Konzepte von CSS	41
3 Die Zukunft von CSS	81
4 Browser-Kompatibilität	97
5 CSS in der Praxis	133
6 Beispiele	271
7 Tools für CSS	355
8 Die CSS-Elemente	391
A Die CD-ROM zum Buch	541
B Die Website zum Buch	543
C Die Referenzkarte	545
D Glossar	547
Index	555

Inhalt

Vorwort	13
1 Einführung in CSS	17
1.1 Von HTML zu CSS	19
1.2 Was sind CSS?	25
1.2.1 Kurze Geschichte der CSS	27
1.3 Wie sehen CSS aus?	28
1.4 »Hallo Welt!« auf CSS	30
1.5 Vorteile und Grenzen des CSS-Einsatzes	39
2 Grundlegende Konzepte von CSS	41
2.1 Selektoren	43
2.1.1 Einfache Elementselektoren	43
2.1.2 Kombinierte Selektoren	47
2.1.3 Class- und ID-Selektoren	47
2.1.4 Kind-Selektoren (CSS2)	49
2.1.5 Folgeelement-Selektoren (CSS2)	51
2.1.6 Attribut-Selektoren (CSS2)	51
2.1.7 CSS3-Selektoren und -Elemente	52
2.1.8 Tabellen-Selektoren	53
2.1.9 Pseudo-Klassen und Pseudo-Elemente	54
2.1.10 Universal-Selektor	55
2.2 Vererbung	57
2.3 Rangfolge und Kaskadierung	58
2.3.1 Die Important-Anweisung	64
2.4 Einbindung von Stylesheets in HTML-Dateien	65
2.4.1 Stil-Anweisungen im HTML-Tag	65
2.4.2 Stil-Anweisungen im Dokumentenkopf	66
2.4.3 Verlinkte Stylesheets	67
2.4.4 Importierte Stylesheets	69
2.4.5 Medienspezifische Stylesheets (CSS2)	70
2.5 Die Kastenform (Box Model)	73
2.6 Benennungen und Werte für Stylesheets	74
2.6.1 Namen für Stylesheets, Klassen und IDs	74
2.6.2 Längen- und Größenangaben	75
2.6.3 Prozentwerte (%)	76

2.6.4	Farben	77
2.6.5	URLs (url)	78
2.6.6	Schlüsselwörter	78
2.7	CSS-Kommentare	78

3 Die Zukunft von CSS 81

3.1	CSS 2.1	83
3.2	CSS 3	83
3.3	XHTML, XML und CSS	85
3.3.1	Von HTML zu X(H)TML	85
3.3.2	XML und CSS	90
3.4	CSS nicht nur im Browser	94
3.4.1	Screenreader	94
3.4.2	PDA's und Mobiltelefone	95

4 Browser-Kompatibilität 97

4.1	Einzelne Browser	99
4.1.1	Mosaic, Netscape Navigator 3.x und früher, Internet Explorer 2.x und früher	99
4.1.2	Netscape Navigator 4.x	99
4.1.3	Netscape Navigator 6 (Mozilla 0.x)	102
4.1.4	Netscape Navigator 7 (Mozilla 1.0)	103
4.1.5	Mozilla 1.8	104
4.1.6	Firefox	104
4.1.7	Internet Explorer 3	105
4.1.8	Internet Explorer 4	105
4.1.9	Internet Explorer 5.x	106
4.1.10	Internet Explorer 6	107
4.1.11	Opera 4	107
4.1.12	Opera 5/6	107
4.1.13	Opera 7	108
4.1.14	Konqueror	108
4.1.15	Safari	109
4.1.16	Lynx	109
4.1.17	Weitere Browser	110
4.2	Browseranteile	110
4.3	Doctypes und Doctype-Switching	112
4.4	Browserweichen und -Hacks	116
4.4.1	@import-Weiche	118
4.4.2	Sternchen-Hack	119
4.4.3	Box Model Hack	120
4.4.4	Vereinfachter Box Model Hack und Erweiterter Vereinfachter Box Model Hack	123
4.4.5	Hochpass-Filter	124

4.4.6	media-Attribut	125
4.4.7	Kommentar-Trick	126
4.4.8	Selektoren-Tricks	126
4.4.9	OperaCatcher	126
4.4.10	IE/Mac-Filter	127
4.4.11	Conditional Comments	128
4.4.12	Browser-Sniffer	129
4.4.13	Hacks selber entwickeln	130

5 CSS in der Praxis 133

5.1	Strategien für den CSS-Designer	135
5.1.1	Valide Dokumente erstellen	135
5.1.2	CSS zur Schriftgestaltung – »Weicher CSS-Einsatz«	137
5.1.3	KomplettlAYOUT mit CSS – »Harter CSS-Einsatz«	139
5.1.4	Entwurfsverfahren für CSS-Sites	144
5.1.5	Fehlersuche in CSS-Dateien	145
5.2	Suchmaschinenoptimierung per CSS	147
5.3	Zugängliche Websites mit CSS (Accessibility)	150
5.4	Schriftgrößen	154
5.4.1	Schlüsselwörter	156
5.4.2	Relative Angaben	158
5.4.3	Punkte	158
5.4.4	Pixel	159
5.4.5	Gemischte Angaben	160
5.4.6	FontSizer (einfach-fuer-alle.de)	163
5.5	CSS-Layouts	164
5.5.1	Positionierungsvarianten	165
5.5.2	CSS-Layouts	175
5.6	Zentrierter Inhalt	193
5.7	Feststehende Bereiche	201
5.7.1	CSS-Frames	208
5.8	CSS-Menüs	213
5.8.1	Dynamische Menüs mit CSS	213
5.8.2	CSS-Menüs mit Listen	224
5.9	Bildergalerie	240
5.10	Schönere Formulare	244
5.11	Druckversion per CSS	254
5.12	Stylesheet-Wechsler	257
5.12.1	Simpler Styleswitcher mit HTML-Mitteln	260
5.12.2	Styleswitcher mit JavaScript	261
5.12.3	Styleswitcher mit PHP	264
5.12.4	JavaScript mit Sicherheitsnetz	267
5.12.5	Browserweiche für Stylesheets	269

6 Beispiele 271

6.1	Aktion Mensch: Einfach für Alle	273
6.1.1	EfA 2001	275
6.1.2	EfA 2003	276
6.1.3	EfA 2004	281
6.1.4	Der Code von Einfach für Alle	282
6.2	Lycos Deutschland	307
6.3	CSS Zen Ocean – ein Beispiellayout für den Zen Garden	335

7 Tools für CSS 355

7.1	CSS-Editoren	357
7.1.1	TopStyle 3	357
7.1.2	Style Studio	361
7.1.3	Cascade 2.0	362
7.1.4	Morphon CSS-Editor	363
7.1.5	Style Master	364
7.1.6	Dreamweaver MX 2004	366
7.1.7	GoLive CS	372
7.2	CSS-Werkzeuge und weitere Tools	377
7.2.1	User-Stylesheets	377
7.2.2	Tidy	382
7.2.3	OpTool	387
7.2.4	Calipers	388
7.2.5	IrfanView	388
7.2.6	Web-Designer-Toolbar	390

8 Die CSS-Elemente 391

8.1	Alphabetische Übersicht	393
8.2	Pseudo-Elemente und Pseudo-Formate	395
8.2.1	:link	395
8.2.2	:visited	398
8.2.3	:hover (CSS2)	399
8.2.4	:focus (CSS2)	400
8.2.5	:active	401
8.2.6	:lang (CSS 2)	402
8.2.7	:first-line	404
8.2.8	:first-letter	406
8.2.9	:first-child (CSS2)	408
8.2.10	:before und :after (CSS2)	409
8.2.11	:not (CSS3)	411
8.3	Inhaltserzeugung	412
8.3.1	content (CSS2, Änderung in CSS 2.1)	412

8.3.2	counter() und counters()	413
8.3.3	counter-increment (CSS2)	416
8.3.4	counter-reset (CSS2)	417
8.3.5	quotes (CSS2)	418
8.4	Schriftformatierungen	420
8.4.1	font-family	420
8.4.2	font-style	423
8.4.3	font-variant	424
8.4.4	font-weight	425
8.4.5	font-size	426
8.4.6	font-size-adjust (CSS, nicht in CSS 2.1)	427
8.4.7	font-stretch (CSS2, nicht in CSS 2.1)	429
8.4.8	font	429
8.4.9	@font-face (CSS2, nicht in CSS 2.1)	431
8.4.10	text-decoration	434
8.4.11	text-shadow (CSS2, nicht in CSS 2.1)	438
8.4.12	text-transform	439
8.4.13	letter-spacing	440
8.4.14	word-spacing (CSS2)	441
8.4.15	white-space	442
8.4.16	line-height	443
8.4.17	text-indent	444
8.4.18	text-align (Änderung in CSS 2.1)	445
8.4.19	vertical-align	446
8.4.20	direction (CSS2)	452
8.4.21	unicode-bidi (CSS2)	453
8.5	Farben und Hintergründe	454
8.5.1	color	454
8.5.2	background-color	455
8.5.3	background-image	456
8.5.4	background-repeat	457
8.5.5	background-attachment	459
8.5.6	background-position (Änderung in CSS 2.1)	460
8.5.7	background	461
8.6	Kastenformatierungen	463
8.6.1	margin	463
8.6.2	padding	465
8.6.3	border-width	467
8.6.4	border-color	468
8.6.5	border-style	469
8.6.6	border	473
8.6.7	width	474
8.6.8	height	475
8.6.9	overflow (CSS2)	477
8.6.10	clip (CSS2, Änderung in CSS 2.1)	478
8.6.11	float	479
8.6.12	clear	483
8.6.13	position (CSS2)	489
8.6.14	box-sizing (CSS3)	491
8.6.15	top (CSS2)	492

8.6.16	right (CSS2)	493
8.6.17	bottom (CSS2)	494
8.6.18	left (CSS2)	495
8.6.19	visibility (CSS2)	496
8.6.20	opacity (CSS3)	497
8.6.21	z-index (CSS2)	498
8.6.22	list-style-type	501
8.6.23	list-style-image	503
8.6.24	list-style-position	504
8.6.25	list-style	504
8.7	Anzeigemodus	505
8.7.1	display (CSS1, Erweiterung in CSS2 – table, Änderung in CSS 2.1, Erweiterung in CSS3 – icon)	505
8.8	Tabellenformatierungen	509
8.8.1	table-layout (CSS2)	509
8.8.2	caption-side (CSS2, Änderung in CSS 2.1)	510
8.8.3	border-collapse (CSS2, Änderung in CSS 2.1)	511
8.8.4	border-spacing (CSS2)	512
8.8.5	empty-cells (CSS2)	513
8.8.6	speak-header (CSS2)	514
8.9	Benutzeroberfläche	515
8.9.1	cursor (CSS2)	515
8.9.2	outline (CSS2)	517
8.9.3	scrollbars (MS-proprietär, kein offizieller CSS-Bestandteil)	518
8.9.4	filter (MS-proprietär, kein offizieller CSS-Bestandteil)	520
8.10	Seitenlayout mit @page (CSS2)	521
8.10.1	size (CSS2, nicht in CSS 2.1)	522
8.10.2	marks (CSS2, nicht CSS 2.1)	523
8.10.3	:left :right :first (CSS2)	524
8.10.4	page-break-before, page-break-after (CSS2)	525
8.10.5	page-break-inside (CSS2)	526
8.10.6	page (CSS2, nicht CSS 2.1)	527
8.10.7	orphans (CSS2)	527
8.10.8	widows (CSS2)	528
8.11	Sprachausgabe	529
8.11.1	speak (CSS2, Änderung in CSS 2.1)	529
8.11.2	volume (CSS2, Änderung in CSS 2.1)	530
8.11.3	speech-rate (CSS2, Änderung in CSS 2.1)	530
8.11.4	pause (CSS2, Änderung in CSS 2.1)	532
8.11.5	cue (CSS2, Änderung in CSS 2.1)	532
8.11.6	play-during (CSS2, Änderung in CSS 2.1)	533
8.11.7	voice-family (CSS2, Änderung in CSS 2.1)	534
8.11.8	pitch (CSS2, Änderung in CSS 2.1)	535
8.11.9	stress (CSS2, Änderung in CSS 2.1)	536
8.11.10	richness (CSS2, Änderung in CSS 2.1)	536
8.11.11	azimuth (CSS2, Änderung in CSS 2.1)	537
8.11.12	elevation (CSS2, Änderung in CSS 2.1)	538
8.11.13	speak-punctuation (CSS2, Änderung in CSS 2.1)	539
8.11.14	speak-numeral (CSS2, Änderung in CSS 2.1)	539

A	Die CD-ROM zum Buch	541
B	Die Website zum Buch	543
C	Die Referenzkarte	545
D	Glossar	547
	Index	555

Vorwort

Cascading Stylesheets haben sich in den vergangenen drei Jahren von einer nur wenigen »Insidern« bekannten, allein mit viel Engagement verwendbaren Methode zum professionellen Mainstream des Webdesign entwickelt.

Was erwartet Sie in diesem Buch?

Immer mehr große Websites, darunter Post.de, GMX.de und Stern.de, haben den Sprung zum tabellenlosen Design gewagt und nutzen die Vorteile dieser neuen Art, Websites zu realisieren.

Bei meiner Beschäftigung mit Stylesheets gab es fünf Phasen (bei HTML war es im Übrigen nicht anders):

1. Vorsichtige Annäherung (»Was ist das überhaupt?«)
2. Naive Begeisterung (»Super! Ich verwende ab sofort nichts anderes mehr!«)
3. Ernüchterung (»Das geht doch alles nicht ...«)
4. Durchkämpfen (»Aha – es geht also doch!«)
5. Routine und Neugier (CSS als Standardansatz und Erforschung der Randbereiche)

Vermutlich wird es bei Ihnen ähnlich sein. Wenn Sie also demnächst einmal vor größeren Schwierigkeiten stehen, dann wissen Sie: Es handelt sich um Phase drei – und Phase vier und fünf müssen bald folgen ...

Dieses Buch will mit einem praxisbezogenen Ansatz die Möglichkeiten, Schwierigkeiten und Grenzen von Cascading Stylesheets anhand konkreter Beispiele aufzeigen (und Ihnen damit helfen, die Phase drei schnell zu überwinden).

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 Einsatz in der Praxis zu beachten?

Das Kapitel 6 ist drei konkreten Beispielen von CSS-gesteuerten Websites gewidmet – zum Nachvollziehen und zum praktischen Einsatz.

Kapitel 7 zeigt Ihnen eine Reihe von Programmen, die beim Erstellen von Stylesheets nützlich sind.

Und schließlich Kapitel 8, 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 Heraus-
nehmen

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 als schnelle Referenz benutzen und für Details dann im Buch nachlesen.

Linkcode

Auf der Webseite zum Buch unter **www.css-praxis.de** finden Sie immer die aktuellen Internetadressen. Hinter jeder URL im Buch finden Sie einen Linkcode. Sollte sich also die Adresse geändert haben, finden Sie mithilfe dieses vierstelligen Codes immer die aktuelle URL. Wenn Sie dennoch feststellen, dass ein Link nicht funktioniert, schicken Sie mir bitte eine E-Mail.

Wie sollten Sie
dieses Buch lesen?

Sie können das Buch ganz normal von vorn nach hinten durchlesen. Ich empfehle dies vor allem, wenn Sie mit Cascading Stylesheets bisher wenig zu tun hatten. Wenn Sie schon etwas über CSS wissen, können Sie auch die Anwendungen ab Kapitel 5, *CSS in der Praxis*, durchgehen und nur im Zweifelsfall die Referenzen des letzten Kapitels zurate ziehen. Oder Sie fangen gleich mit den Beispiel-Websites in Kapitel 6, *Beispiele*, an und sehen, wie es andere gemacht haben.

Icons und
Hinweise

Um Sie auf bestimmte Dinge aufmerksam zu machen, werden einige Icons in der äußeren Spalte verwendet:



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



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 (der gleich ein ganzes Kapitel beigesteuert hat), Daniel Ludwin, Ernesto Gimenez, Kevin Smith, 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!

Vielen Dank an meinen Lektor Stephan Mattescheck, der mich nicht nur auf die Idee zu diesem Buch brachte, sondern mich auch bei der Erstellung unterstützte und insbesondere viel Geduld 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, Dave Shea (für Csszengarden), 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 musste und mir außerdem als fachkundige Lektorin zur Seite stand.

Kai Laborenz

Berlin, Dezember 2004

2 Grundlegende Konzepte von CSS

Bevor wir uns mit den einzelnen Elementen befassen, möchte ich die fünf grundlegenden Konzepte von Stylesheets erläutern: Selektoren, Vererbung, Rangfolge und Kaskadierung, Einbindung von Stylesheets sowie Werte für Stylesheets.

2.1 Selektoren

Die so genannten Selektoren (siehe Abbildung 1.8) regeln, wie die Stileigenschaften Klassen oder HTML-Elementen zugeordnet werden. Dabei nimmt man zunächst den Namen des Tags, für den der Selektor stehen soll, und setzt dann die Eigenschaft in geschweifte Klammern (`{}`). Hinter dem Doppelpunkt können ein oder mehrere Werte stehen:

```
Selektor {Eigenschaft: Wert1; Eigenschaft: Wert2;
          Eigenschaft: Wert3; ... }
```

2.1.1 Einfache Elementselektoren

Die einfachste Möglichkeit, Stile einem HTML-Dokument zuzuordnen, besteht in der Zuweisung zu bestimmten HTML-Tags. Zum Beispiel wird mit `p {color: black}` allen HTML-Elementen `<p>` die Farbe Schwarz zugewiesen.

Wenn Sie mehreren Elementen den gleichen Stil zuweisen möchten, notieren Sie die Elemente nacheinander durch Kommata getrennt:

CSS-Stile den
HTML-Tags
zuweisen

```
p, h1, ol {color: black;}
```

Mehrere Stil-Anweisungen für ein Element werden durch Semikola getrennt:

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

Sie können beide Methoden auch kombinieren:

```
p, h1, ol {color: black; background-color: green; }
```

Hier werden den HTML-Tags `<p>`, `<h1>` und `<ol>` die Eigenschaften *Schriftfarbe: Schwarz* und *Hintergrundfarbe: Grün* zugewiesen.

Die Aufzählung mehrerer Selektoren unterscheidet sich nur durch die Kommata von der Definition kontextabhängiger Selektoren (siehe nächsten Abschnitt), sodass hier leicht Fehler entstehen können.

Spezial-Tags für CSS Was aber tun Sie, wenn Sie einen Bereich innerhalb eines HTML-Elements formatieren möchten oder einen Bereich, der mehrere HTML-Elemente umfasst?

Für diesen Fall sieht HTML ab der Version 4 zwei besondere Tags für die Arbeit mit Stilen vor:

`<span>` und `<div>`.

Diese Tags haben allein keine direkten Auswirkungen auf ein Dokument und wirken erst durch die Kombination mit Stylesheets. Eine hohe Flexibilität bei der Zuweisung von Stilen lässt sich durch die beiden HTML-Tags zusammen mit den Stil-Selektoren `class` und `id` erreichen.

Während `<div>` ein so genanntes Block-Element definiert (wie z.B. `<p>` oder `<h1>`), lassen sich mit `<span>` beliebige Bereiche innerhalb von Blocks oder über mehrere Blocks hinweg ansprechen (natürlich muss trotzdem die korrekte Verschachtelung von Tags beachtet werden – siehe dazu auch Abschnitt 5.1.1, *Valide Dokumente erstellen*).

Ein Beispiel:

Neue HTML-Tags: span und div Sie wollen ein Wort in einem Absatz herausstellen:

```
<p>Dies ist ein normaler Absatz. Er enthält mehrere Sätze  
und ein Wort ist <span class="wichtig">besonders</span>  
wichtig. Dieses Wort wird durch CSS formatiert.</p>
```

Hier ist das Wort »besonders« durch den `<span>`-Tag markiert. Noch passiert nichts Besonderes. Erst wenn Sie die Klasse `.wichtig` in einem Stylesheet definieren, erhält das Wort »besonders« eine spezielle Formatierung (hier: vergrößerte Schrift und Farbe Rot).

```
.wichtig { font-size: 120%;  
          color: #ff0000; }
```

Das nächste Beispiel zeigt eine mit SPAN und DIV gestaltete Seite:

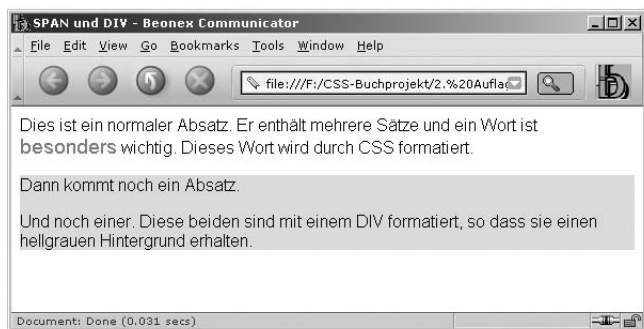


Abbildung 2.1 Formatierung mit SPAN und DIV

Der vollständige Code sieht so aus:

```

1:  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
    Transitional//EN">
2:  <html>
3:  <head>
4:    <title>SPAN und DIV</title>
5:    <style type="text/css">
6:    <!--
7:      .wichtig {
8:        font-size: 120%;
9:        color: #ff0000;
10:     }
11:     .hintergrund {
12:       background-color: #dddddd;
13:     }
14:    -->
15:  </style>
16: </head>
17: <body>
18:   <p>Dies ist ein normaler Absatz. Er enthält
    mehrere Sätze und ein Wort ist
    <span class="wichtig">
    besonders</span> wichtig. Dieses Wort wird durch
    CSS formatiert. </p>
19:   <div class="hintergrund">
20:     <p>Dann kommt noch ein Absatz.</p>
21:     <p>Und noch einer. Diese beiden sind mit einem
    DIV formatiert, sodass sie einen hellgrauen

```

```

                Hintergrund erhalten. </p>
22:      </div>
23:    </body>
24:  </html>

```

Listing 2.1 Formatierung mit SPAN und DIV

Kompatibilitäts- tabellen

Als Nächstes sehen Sie eine Kompatibilitätstabelle, die anzeigt, wie das in diesem Abschnitt beschriebene Element von den aktuellen (und nicht mehr ganz so aktuellen) Browsern interpretiert wird. Leider werden immer noch viele CSS-Elemente nicht von allen Browsern so angezeigt, wie es nach der Spezifikation sein sollte. Solchen Tabellen werden Sie in den folgenden Kapiteln noch öfter begegnen – sie sollen Ihnen einen schnellen Überblick über die Praxissituation des besprochenen Themas geben. Ergänzend liegt dem Buch eine farbige Browserreferenzkarte bei, die alle Elemente und Eigenschaften in der Übersicht zeigt.

Kompatibilität

	NN 4	NN 7	Mz 1.4	Mz 1.8/FF	IE 5	IE 6	O5	O6	O7	Sf 1.2	Konq
PC	j	j	j	j	j	j	j	j	j	–	–
Mac/Linux	j	j	j	j	j	–	j	–	–	j	j

Erklärung der in dieser und in den folgenden Tabellen verwendeten Symbole:

- j = Ja, wird unterstützt
- n = Nein, wird nicht unterstützt
- t = wird teilweise unterstützt
- f = Unterstützung ist fehlerhaft
- NN = Netscape Navigator
- Mz = Mozilla und kompatible Browser
- FF = Firefox (Mozilla und Firefox verwenden die gleichen Bibliotheken zur Anzeige)
- IE = Internet Explorer
- O = Opera
- Konq = Konqueror

Die erste Zeile der Ergebnisse bezieht sich auf die Windows-Version, die zweite beschreibt das Verhalten von Macintosh-Systemen – im Fall von Konqueror das unter Linux.

Die Zahlen hinter den Buchstaben weisen auf zusätzliche Erläuterungen hin, die im Abschnitt »Bemerkungen« jeweils aufgeführt werden.

2.1.2 Kombinierte Selektoren

Die obigen Anweisungen beziehen sich immer auf alle im Geltungsbereich des Stylesheets vorkommenden Instanzen des angegebenen Elements, d. h., mit `p {color: black;}` bekommen Absätze dieses Format zugewiesen. Es ist aber auch möglich, nur einer bestimmten Kombination von Elementen einen Stil zuzuweisen:

```
h1 {color: black;}
em {color: red;}
```

würde allen Überschriften erster Ordnung `<h1>` die Farbe Schwarz und allen `<em>`-Elementen die Farbe Rot zuweisen. Wenn nun aber nur die `<em>`-Elemente *innerhalb* von `<h1>`-Elementen gefärbt werden sollen, kann das so ausgedrückt werden:

```
h1 {color: black;}
h1 em {color: red;}
```

Mit dem ersten Befehl wird allen `<h1>`-Elementen die Textfarbe Schwarz zugewiesen. In der zweiten Zeile werden dann die `<em>`-Elemente, die sich innerhalb eines `<h1>`-Elementes befinden, rot gefärbt.

Kompatibilität

	NN 4	NN 7	Mz 1.4	Mz 1.8/ FF	IE4	IE 5	IE 6	O5	O6	O7	Sf 1.2	Konq
PC	f	j	j	j	j	j	j	j	j	j	–	–
Mac/Linux	f	j	j	j	j	j	–	j	j	–	j	j

2.1.3 Class- und ID-Selektoren

CSS bietet zusätzliche Möglichkeiten, Elemente zu formatieren: die Selektoren CLASS und ID.

Jedes HTML-Element unterhalb des `<body>`-Elements kann mit einer Klasse versehen werden. So können unterschiedlichen HTML-Tags mit Hilfe einer Klasse Formatierungen zugeordnet werden.

Klassendefinitionen beginnen immer mit einem Punkt.

```
.wichtig {font-weight: bold; color: red;}
```

definiert eine Klasse »wichtig«.

Mit der Notation `<p class="wichtig"> ... </p>` oder `<h1 class="wichtig"> ... </h1>` können diesen oder beliebigen anderen HTML-Tags die für »wichtig« festgelegten Eigenschaften zugewiesen werden.

class und id

Im Gegensatz zu Klassen bezeichnen IDs immer dokumentenweit einzigartige Elemente. So können Anweisungen für jedes einzelne Element eines HTML-Dokumentes vorgenommen werden.

Definitionen für ID-Selektoren beginnen immer mit einem Doppelkreuz (`>#<`), können aber ansonsten nahezu beliebige Bezeichnungen tragen.

```
#hauptmenue {background-color: red}
```

definiert für die ID »hauptmenue« die Hintergrundfarbe des Elements als »rot«. Mit `<div id="hauptmenue"> ... </div>` wird ein Bereich des HTML-Dokumentes als »Hauptmenue« definiert. Sie können einem Element eine ID und eine Klasse gleichzeitig zuweisen. Was passiert, wenn dem Element dadurch widersprüchliche Anweisungen zugewiesen werden? Das regelt die Rangfolge der CSS-Eigenschaften, die Sie in Abschnitt 2.3, *Rangfolge und Kaskadierung*, kennen lernen.

Kompatibilität

	NN 4	NN 7	Mz 1.4	Mz 1.8/FF	IE4	IE 5	IE 6	O5	O6	O7	Sf 1.2	Konq
PC	f ¹	j	j	j	j	j	j	j	j	j	–	–
Mac/Linux	f ¹	j	j	j	j	j	–	j	j	–	j	j

1 Der Netscape Navigator 4.x unterstützt keine ID-Selektoren der Form `p#id`.

Beachten Sie, dass bei Klassen und IDs im HTML-Code nur die Bezeichnungen angegeben werden, nicht aber Punkt oder Doppelkreuz. Außerdem sollten Sie weder Umlaute in den Namen verwenden noch

darf das erste Zeichen eine Zahl sein. Am besten, Sie verwenden nur die kleinen Buchstaben des ASCII-Zeichensatzes. Mehr zur Schreibweise von CSS-Anweisungen finden Sie in Abschnitt 2.6, *Benennungen und Werte für Stylesheets*.

2.1.4 Kind-Selektoren (CSS2)

Mit dem Kind-Selektor (`child`) werden Elemente angesprochen, die im Sinne objektorientierter Programmierung »Kinder« eines vorgenannten Selektors sind.

In der folgenden Abbildung 2.2 sind z.B. die Elemente HEAD und BODY Kinder von HTML, während B und IMG Kinder des unteren P sind.

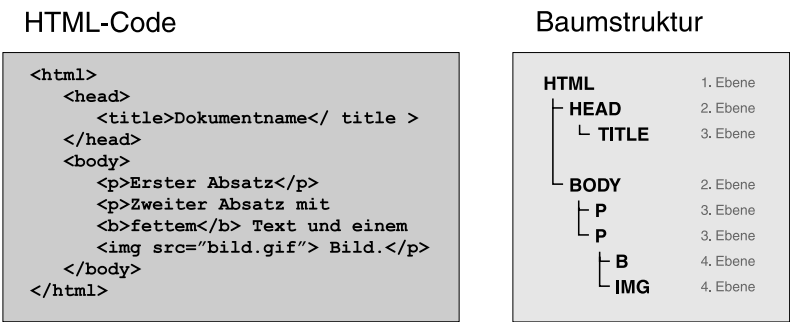


Abbildung 2.2 Baumstruktur eines HTML-Dokumentes

```
body>p {
  line-height: 1.8em;
  background-color: #dedede
}
```



Dies setzt für alle Absätze, die Kinder des `<body>`-Elementes sind, die Zeilenhöhe auf das 1,8-fache der Schriftgröße und die Hintergrundfarbe auf ein helles Grau. Im Unterschied zu

```
body p {border: 1px dashed black}
```

werden hier keine Absätze angesprochen, die zwischen `<body>` und `<p>` noch einen weiteren HTML-Tag ausweisen. Das Beispiel zeigt die Anwendung: Während die Anweisung `body p` alle Absätze mit einem unterbrochenen Rahmen versieht, wirkt `body>p` nur auf die beiden äußeren Absätze, die direkte Kinder von `<body>` sind, und färbt deren Hintergrund hellgrau:

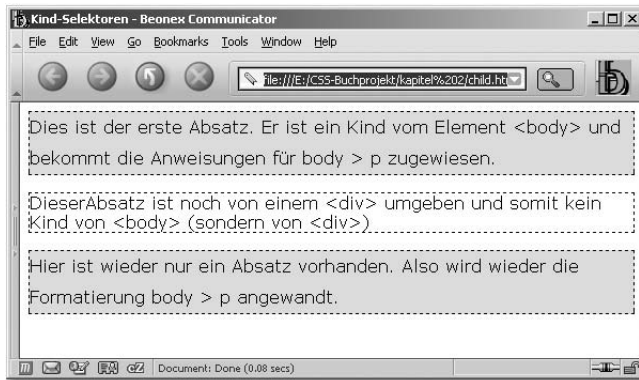


Abbildung 2.3 Anwendung des Kind-Selektors

```

1:  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
    Transitional//EN">
2:  <head>
3:    <title>Kind-Selektoren</title>
4:    <style type="text/css">
5:    <!--
6:      body>p {
7:        line-height: 1.8em;
8:        background-color: #dedede
9:      }
10:     body p {
11:       border: 1px dashed black
12:     }
13:    -->
14:  </style>
15: </head>
16: <body>
17:   <p>Dies ist der erste Absatz. Er ist ein Kind vom
      Element &lt;body> und bekommt die Anweisungen
      f&uuml;r body &gt; p zugewiesen.</p>
18:   <div>
19:     <p>Dieser Absatz ist noch von einem
      &lt;div> umgeben und somit kein Kind
      von &lt;body> (sondern von &lt;div>)</p>
20:   </div>
21:   <p>Hier ist wieder nur ein Absatz vorhanden.
      Also wird wieder die Formatierung

```

```
body &gt; p angewandt.</p>
22: </body>
23: </html>
```

Listing 2.2 Kind-Selektoren

Kompatibilität

	NN 4	NN 7	Mz 1.4	Mz 1.8/FF	IE4	IE 5	IE 6	O5	O6	O7	Sf 1.2	Konq
PC	n	j	j	j	n	n	n	j	j	j	–	–
Mac/Linux	n	j	j	j	n	j	–	j	j	–	j	j

2.1.5 Folgeelement-Selektoren (CSS2)

Ein Folgeelement-Selektor markiert ein Element, das unmittelbar auf ein anderes folgt und mit diesem dasselbe Elternteil hat.

```
h1+p {margin-top: -5px;}
```

Dies sorgt dafür, dass der obere Abstand eines Absatzes auf –5 Pixel gesetzt wird, wenn er direkt auf eine <h1>-Überschrift folgt.

Kompatibilität

	NN 4	NN 7	Mz 1.4	Mz 1.8/FF	IE4	IE 5	IE 6	O5	O6	O7	Sf 1.2	Konq
PC	n	j	j	j	n	n	n	j	j	j	–	–
Mac/Linux	n	j	j	j	n	j	–	j	j	–	j	j

2.1.6 Attribut-Selektoren (CSS2)

CSS2 erlaubt es, Elemente aufgrund ihrer Attribute anzusprechen.

```
<a href="http://www.w3c.org" target="_top">
```

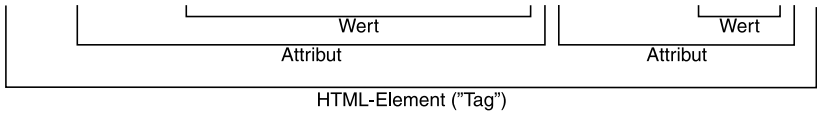


Abbildung 2.4 Das HTML-Element und seine Attribute

```
width: 300px;  
width: 200px;  
}
```

Dabei wird ein weiterer Fehler des Internet Explorers genutzt: Die Syntax »* html« bedeutet laut CSS-Spezifikation »*ein beliebiges Element, das von dem HTML-Element gefolgt wird*«. Da das HTML-Element immer das erste Element eines HTML-Dokuments ist, kann diese Regel nie zutreffen – dafür notierte Anweisungen werden von einem korrekt arbeitenden Browser ignoriert.

Der IE ignoriert aber das »*«, wenn es vor dem »html« steht, und führt also die Anweisungen aus: zunächst die erste Zeile »width: 300px« für alle Internet Explorer und dann die zweite Zeile aufgrund des enthaltenen »\« nur für IE6.

Der Vorteil gegenüber dem Vereinfachten Box Model Hack: Hiermit können auch Anweisungen bearbeitet werden, deren erster Buchstabe a, b, c, d, e oder f ist. Außerdem sorgt die Verwendung des »*« dafür, dass alle Browser abgesehen vom fehlerhaften Internet Explorer die Regel mit den Escape-Zeichen (»\«) ignorieren. Weitere Maßnahmen für Netscape 5 oder Opera 5 sind also unnötig. Der Hack ist zukunftsfähig – wenn der Internet Explorer in einer neuen Version den »*«-Fehler nicht mehr aufweist, ergeben sich keine negativen Nebeneffekte.

Die zahlreichen Varianten des Box Model Hacks sind der Diskussionsliste *css-discuss* zu entnehmen:

<http://css-discuss.incutio.com/?page=BoxModelHack> (Linkcode 0032)

4.4.5 Hochpass-Filter

Ebenfalls aus der Feder von Tantek Çelik stammt der folgende Hack, mit dem ältere Browser (ganz genau: solche, die Section 7.1 der offiziellen CSS1 Test Suite nicht bestehen) vom Laden eines Stylesheets abgehalten werden. Das Stylesheet wird dann nur von diesen Browsern geladen:

- ▶ IE5/Mac oder höher
- ▶ IE6/Windows oder höher
- ▶ Netscape 6 oder höher
- ▶ Opera 5 oder höher

- Im aufrufenden HTML-Dokument wird ganz normal ein Stylesheet verlinkt:

```
<link rel="stylesheet" href="hochpass.css" />
```

- In diesem Stylesheet befindet sich folgende Syntax:

```
@import "null?\"";
@import "richtiges_stylesheet.css";
```

Durch die schon aus dem Box Model Hack bekannte Zeichenkombination `\"` werden ältere Browser aus dem Gleichgewicht gebracht. Sie importieren das erste Stylesheet `"null?\"`, welches keine Anweisungen enthält, und sehen dann diese Syntax vor sich:

```
{ ":@import" richtiges_stylesheet.css "};
```

Da dies keine gültige CSS-Deklaration oder Anweisung ist, wird der Teil ignoriert.

Moderne Browser sehen einfach zwei zu importierende Stylesheets, die nacheinander geladen werden. Da das erste leer ist, kommt nur `richtiges_stylesheet.css` zur Anwendung.

Um diese Lösung zu validieren, müssen Sie tatsächlich über ein Stylesheet mit dem Dateinamen *»null«* mit einer Zeile Text (z.B. einem Kommentar) auf dem Server verfügen.

4.4.6 media-Attribut

Der Netscape Navigator 4.x und einige andere Browser haben Schwierigkeiten, Stylesheets anhand ihrer `media`-Zuweisung zu verarbeiten. Netscape 4.x erkennt nur Stylesheets mit der Zuweisung `media="screen"`. Das führt uns zu zwei weiteren Möglichkeiten, Stil-Anweisungen aufzusplitten:

Schreibweise	Versteckt das Stylesheet vor ...
<code><link href="kein_nn4x.css" type="text/css" rel="stylesheet" media="all"/></code>	Netscape 4.x
<code>@media all { ... }</code>	Netscape 4.x Internet Explorer 4 (Win) Internet Explorer 4.01 (Mac) Internet Explorer 5.0 (Mac)

Tabelle 4.5 media-Attribut

4.4.7 Kommentar-Trick

Wenn Sie einen Kommentar direkt hinter einen Selektor platzieren, ignorieren bestimmte Browser die dann folgenden Werte.

Mit der folgenden Schreibweise würde also die Angabe `width: 250px` von den angegebenen Browsern nicht ausgeführt.

Schreibweise	Versteckt die Stile vor ...
<code>#inhalt/* */ width: 250px</code>	Internet Explorer 5 (Win) und älter Internet Explorer 4.01 (Mac) Internet Explorer 5.0 (Mac)

4.4.8 Selektoren-Tricks

Selektoren werden momentan nur von wenigen Browsern interpretiert und eignen sich daher auch zum selektiven Anwenden von Stil-Anweisungen (mehr zum Thema Selektoren finden Sie in Abschnitt 2.1). Wir verwenden hier den Attribut-Selektor und den Kind-Selektor als Browserweiche.

Schreibweise	Versteckt die Stile vor ...
<pre>div[id] { ... }</pre> Damit werden alle <code><div></code>s angesprochen, die eine ID besitzen.	Netscape 4.x Internet Explorer 6 (Win) und älter Internet Explorer 4.01 (Mac) Internet Explorer 5.0 (Mac) Opera 3.6 iCab 2.51 (Mac) Leider wendet der iCab die Formatierung einfach auf alle <code><div></code> s an, statt sie zu ignorieren.
<pre>div>p { ... }</pre> Dies markiert alle Absätze, die Kinder von <code><div></code>s sind.	Netscape 4.x Internet Explorer 6 (Win) und älter Internet Explorer 4.01 (Mac) Internet Explorer 4.5 (Mac) iCab 2.51 (Mac)

Tabelle 4.6 Selektoren-Tricks

4.4.9 OperaCatcher

Es gibt ein paar Situationen, in denen Anweisungen vor Opera-Browsern versteckt werden sollen (Abschnitt 5.8.1 zeigt eine davon).

Da Opera die Pseudo-Klasse `:first-child` nicht interpretiert, lassen sich beliebige Anweisungen mit folgender Syntax vor Opera verstecken:

```
head:first-child+body #name { ... }
```

versteckt alle Angaben für Elemente mit der ID `name` vor Opera.

Wie funktioniert dies?

`head:first-child+body` ist eigentlich eine sinn- und folgenlose Anweisung. Da das Element `<head>` immer das erste Element seines Eltern-Elements (und zwar `<html>`) ist und ihm auch immer `<body>` folgt, trifft dieser Selektor immer auf alle Elemente eines HTML-Dokumentes zu. In der Kombination mit `#name` wird dann ein Element `#name` selektiert, das sich innerhalb von `<body>` befindet (was immer zutrifft, wenn `<div id="name"> ... </div>` irgendwo im Dokument verwendet wird). Letztendlich ist

```
head:first-child+body #name { ... }
```

gleichbedeutend mit

```
#name { ... }
```

Nur wird es von Opera eben nicht interpretiert. Da allerdings auch der Internet Explorer (bis Version 6) mit dieser Anweisung nicht klar kommt, müssen Sie – um wirklich nur Opera auszuschließen – diese Anweisung noch vor dem Internet Explorer verstecken, z.B. mit einem der vorher genannten Tricks.

4.4.10 IE/Mac-Filter

Der Internet Explorer 5 für Macintosh-Rechner verhält sich in vielen Situationen anders als sein Windows-Pendant und auch alle anderen Browser. Daher ist es sehr nützlich, ihn isoliert ansprechen zu können. Dies funktioniert in beide Richtungen:

```
/* ohne IE/Mac */
#keinMacIE { ... }
/**/
```

Hier werden die zwischen den beiden Kommentarzeilen liegenden Anweisungen vom IE/Mac nicht ausgeführt.

```
63: </body>
64: </html>
```

Listing 5.18 Listing: CSS-Frames für Internet Explorer unter besonderer Berücksichtigung des IE5/Mac

Der Internet Explorer muss sich im Quirks-Modus befinden, damit dieses Beispiel funktioniert.



5.8 CSS-Menüs

5.8.1 Dynamische Menüs mit CSS

Nachdem wir Menüs schon per CSS platziert haben, wollen wir jetzt einmal sehen, wie wir sie mit CSS etwas aufpeppen können.

Rollover-Menüs

Recht beliebt sind Effekte, bei denen vor den Menüpunkten ein Anzeiger je nach Mausposition die Position des Zeigers anzeigt. Normalerweise wird dies mit Bildern vorgenommen, die per JavaScript ausgetauscht werden.

Mit CSS geht das aber auch – sogar recht einfach – und zudem ohne den Einsatz von JavaScript oder sogar ohne Grafiken (die immer negativ auf die Ladezeit wirken).

Die Schlüssel für ein solches CSS-Menü sind die Pseudo-Klasse `:hover` und die Eigenschaft `visibility`.

Wir verwenden einen Bereich, dessen Sichtbarkeit eingangs auf »nicht sichtbar« (`visibility: hidden`) gesetzt ist und der – mit allen seinen Inhalten – daher nicht angezeigt wird. Erst beim Überfahren mit der Maus verändern wir seine Sichtbarkeit auf `visibility: visible` und zeigen ihn damit an.

In diesen Bereich können Sie ein Aufzählungszeichen setzen. HTML bietet seit der Version 4.0 eine Reihe von Sonderzeichen, die sich auch als Menüanzeiger eignen:

Zeichen	HTML-Code	Unicode
»	<code>&raquo;</code>	187
→	<code>&rarr;</code>	8594

Tabelle 5.2 Geeignete Zeichen zur Verzierung von Menüeinträgen

Zeichen	HTML-Code	Unicode
⇒	⇒	8658
•	•	149, 8226
#	&lolz;	9674

Tabelle 5.2 Geeignete Zeichen zur Verzierung von Menüeinträgen (Forts.)

Sie können natürlich auch völlig auf die Anzeige eines Zeichens verzichten und die Hintergrundfarbe des Bereichs ändern.

Sehen wir uns einmal ein Beispiel an:



```

1:  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
2:  xhtml1-transitional.dtd">
3:  <html xmlns="http://www.w3.org/1999/xhtml" lang=de>
4:  <head>
5:    <title>Checkliste</title>
6:    <style type="text/css">
7:      /*<![CDATA[*]
8:      body {
9          background-color: #fcfcfc;
10:         color: #000;
11:     }
12:     h1 {
13:         margin: 0 2%;
14:         font-size: 1.4em
15:     }
16:     p {
17:         margin: 1em 2%;
18:         font-size: 0.8em;
19:         line-height: 140%;
20:     }
21:     p a{text-decoration: none; }
22:     a:link, a:visited { color: #000}
23:     a:hover {
24:         color: #000;
25:         background: #dedede;
26:     }

```

Bis hierhin enthält das Stylesheet nichts Ungewöhnliches. Wir definieren die Bereiche `<h1>` und `<p>` sowie Links und sorgen dafür, dass Links in Absätzen (`<p>`) nicht unterstrichen sind.

```
27:  a .bullet {
28:      visibility: hidden;
29:      background-color: #c00;
30:      text-decoration: none;
31:  }
```

Hier kommt unser versteckter Bereich. Für ihn legen wir eine Hintergrundfarbe (Dunkelrot) fest und definieren ihn als versteckt.

```
32:  .item {
33:      text-decoration: underline;
34:  }
```

Diese Anweisung benötigen wir, da Mozilla keine nicht unterstrichenen Texte innerhalb unterstrichener Texte akzeptiert (siehe Abschnitt 8.4.10, *text-decoration*). Dadurch wird auch der HTML-Code etwas komplizierter, aber dazu gleich mehr.

```
35:  a:hover .bullet{
36:      visibility: visible;
37:  }
```

Und hier kommt die entscheidende Anweisung. Beim Überfahren mit der Maus (`:hover`) setzen wir den Bereich `.bullet` (genauer: `.bullet`, das sich innerhalb eines Links befindet) auf sichtbar.

```
38:  /*]]>*/
39:  </style>
40: </head>
41: <body>
42:   <h1>Checkliste für den Urlaub</h1>
43:   <p>
44:     <a href="#1">
45:       <span class="bullet">&nbsp;&nbsp;&nbsp;</span>&nbsp;&nbsp;&nbsp;
46:       <span class="item">Sonnencreme</span>
47:     </a>
```

Der einzelne Menüeintrag besteht aus dem Link `<a href="..."> ... </a>` und zwei eingebundenen Bereichen (jeweils durch `<span>` markiert). Der erste enthält den unsichtbaren Bereich, den wir beim Überfahren sichtbar machen wollen, der zweite den eigentlichen Link.

Warum diese etwas umständliche Notation für den zweiten Link?

Standardmäßig ist die Unterstreichung von Links immer aktiviert. Daher wird der gesamte Link zunächst einmal unterstrichen. Wir haben zwar im Stylesheet die Unterstreichung für den Bereich `.bullet` abgeschaltet, aber leider unterstreicht Mozilla immer den gesamten Link – egal, was für untergeordnete Bereiche definiert wird. Daher müssen wir mit `p a {text-decoration: none }` zunächst die Unterstreichung für den ganzen Absatz ausschalten und sie dann für den Textteil des Links (nämlich die Klasse `.item`) wieder einschalten. Sie könnten diese Konstruktion nur umgehen, indem Sie die Link-Unterstreichung generell ausschalten – was aber aus Gründen der Benutzerfreundlichkeit nicht anzuraten ist.

```
48:      <br />
49:      <a href="#2">
50:          <span class="bullet">&nbsp;&nbsp;&nbsp;</span>&nbsp;&nbsp;&
51:          <span class="item">Bücher</span>
52:      </a><br />
53:      <a href="#15">
54:          <span class="bullet">&nbsp;&nbsp;&nbsp;</span>&nbsp;&nbsp;&
55:          <span class="item">Kreuzworträtsel</span>
56:      </a><br />
57:      <a href="#3">
58:          <span class="bullet">&nbsp;&nbsp;&nbsp;</span>&nbsp;&nbsp;&
59:          <span class="item">Spiele</span>
60:      </a><br />
61:      <a href="#4">
62:          <span class="bullet">&nbsp;&nbsp;&nbsp;</span>&nbsp;&nbsp;&
63:          <span class="item">Lange Klamotten</span>
64:      </a><br />
65:      <a href="#5">
66:          <span class="bullet">&nbsp;&nbsp;&nbsp;</span>&nbsp;&nbsp;&
67:          <span class="item">Feste Schuhe</span>
68:      </a>
69:  <p>
70: </body>
71: </html>
```

Listing 5.19 Dynamische Menüs – Rollover

Das Ganze sieht dann folgendermaßen aus:



Abbildung 5.25 Das Menü im »Ruhezustand« ...



Abbildung 5.26 ... und in Aktion (mit geändertem Hintergrund) ...



Abbildung 5.27 ... oder mit einem Aufzählungszeichen.

Reine CSS-Menüs mit mehreren Ebenen

Nach der Spezifikation zu CSS2 ist es möglich, dynamische Pseudo-Klassen (:hover, :active, :focus) auch anderen Elementen als Links (<a>) zuzuordnen. Mozilla 1.0 war der erste Browser, der diese Vorgaben auch umsetzt. Damit ergeben sich neue Möglichkeiten, dynami-

sche Menüs ganz ohne JavaScript herzustellen. Die folgenden Beispiele entstanden nach einer Technik, die Eric Meyer erstmals beschrieben hat.



Die beschriebene Technik funktioniert momentan nur in Mozilla (ab 1.0) und darauf aufbauenden Browsern wie Firefox und in Opera 7. Weder der Internet Explorer (bis Version 6) noch Opera (vor Version 7) können mit `:hover` für andere Elemente als Links etwas anfangen. Da die Beispiele den CSS-Spezifikationen entsprechen, ist zu hoffen, dass folgende Browserversionen auch damit zurechtkommen. Mit etwas JavaScript kann man aber auch dem Internet Explorer das »hovern« beibringen. Dazu mehr im Abschnitt über CSS-Listen (Abschnitt 5.8.2).

Im Übrigen ist der Code abwärtskompatibel: Ältere CSS-Browser sehen immerhin die erste Menüebene mit funktionsfähigen Links und Nicht-CSS-Browser erhalten den ganzen Menübaum unformatiert angezeigt. Am Ende des Kapitels stelle ich noch einen Trick vor, der die Abwärtskompatibilität weiter verbessert.

Die erste Variante erstellt ein seitlich aufklappendes Menü, wie es auf vielen Websites zu sehen ist.

Der Trick besteht darin, einen Kasten mit der zweiten Menüebene anzulegen und seinen Anzeigemodus auf `display: none` (Element wird nicht angezeigt) zu setzen. Wenn nun die Maus über den entsprechenden Menüpunkt in der ersten Menüebene streift, greift die Pseudo-Klasse `:hover`. Für diese Pseudo-Klasse definieren wir, dass der Anzeigemodus auf `display: block` gesetzt wird, und das Element wird sichtbar!



Abbildung 5.28 Dynamisch aufklappendes Menü

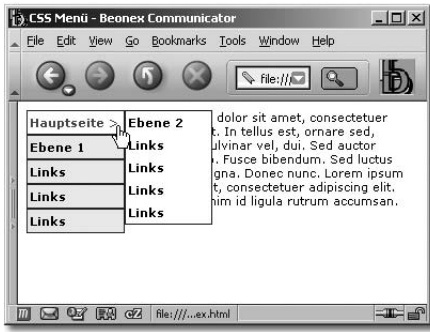


Abbildung 5.29 Beim Überfahren mit der Maus erscheint die zweite Ebene

Sehen wir uns den Code im Detail an:

```

1:  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
    xhtml1-transitional.dtd">
2:  <html xmlns="http://www.w3.org/1999/xhtml" lang=de>
3:  <head>
4:    <title>CSS Men&uuml;</title>
5:    <style type="text/css">
6:      /*<![CDATA[* /
7:      body {
8:        font: bold 11px Verdana,Arial;
9:      }
10:   #menu1 {
11:     width: 90px;
12:     float: left;
13:   }

```

Dies definiert die Breite des Menüs und legt fest, dass es linksbündig angeordnet wird.

```

14:   .menulitem {
15:     background: #efefef;
15:     border: 1px solid #800000;
17:     color: #000;
18:     font: bold 11px Verdana,Arial;
19:     text-decoration: none;
20:     width: 90px;
21:   }
22:   .menulitem a {

```

```

23:      display: block;
24:      text-decoration: none;
25:      color: #000;
26:      padding: 4px 2px;
27:  }

```

Diese beiden Styles legen das Aussehen der einzelnen Menüeinträge fest. Das `display: block` sorgt dafür, dass die Links (eigentlich eingebundene Elemente) als Block-Level-Elemente behandelt werden.

```

28:  #menu2 {
29:      position: absolute;
30:      top: 8px;
31:      left: 100px;
32:      display: none;
33:      background: #fff;
34:      border: 1px solid #000080;
35:      color: #000;
36:      font: bold 11px Verdana,Arial;
37:      text-decoration: none;
38:      width: 80px;
39:      z-index: 20;
40:  }

```

Dies ist der Kasten für die zweite Menüebene. Wichtig sind die ersten Anweisungen: `position: absolute`, und die folgenden Angaben `top` und `left` erlauben es uns, den Kasten beliebig zu positionieren. `display: none` sorgt dafür, dass er jetzt noch nicht angezeigt wird. Und der letzte Wert zum `z-index` legt das Menü über andere Seitenelemente.

```

41:  div.menuitem a:hover {
42:      background: #fff;
43:      color: #cc0000;
44:  }

```

Hiermit ändern Sie die Farben der Menüeinträge der ersten Menüebene.

```

45:  div.menuitem:hover #menu2 {
46:      display: block;
47:  }

```

Dies ist der Teil, der unsere zweite Menüebene zum Vorschein bringt. Für den Kasten `#menu2`, der sich innerhalb eines `div`s der Klasse `.menu-item` befindet, über dem der Mauszeiger schwebt, wird `display: block` definiert – der Kasten wird angezeigt.

```

48:    /*]]>*/
49:    </style>
50: </head>
51: <body>
52:   <div id="menu1">
53:     <div class="menulitem">
54:       <a href="index.html">Hauptseite </a>
55:       <div id="menu2">
56:         <a href="#">Ebene 2</a>
57:         <a href="#">Links</a>
58:         <a href="#">Links</a>
59:         <a href="#">Links</a>
60:         <a href="#">Links</a>
61:       </div> /* Ende menu2 */
62:     </div> /* Ende menuitem */

```

Der gesamte Kasten `menu2`, der die zweite Menüebene enthält, befindet sich innerhalb des ersten Menüpunkts der ersten Menüebene – die logische Zuordnung des Submenüs zum entsprechenden Punkt des Hauptmenüs ist also gegeben.

```

63:   <div class="menulitem"><a href="#">Ebene 1
64:     </a></div>
65:   <div class="menulitem"><a href="#">
66:     Links</a></div>
67:   <div class="menulitem">
68:     <a href="#">Links</a></div>
69:   <div class="menulitem">
70:     <a href="#">Links</a></div>
71: </div> /* Ende menu1 */
72: Lorem ipsum dolor sit amet, consectetur
73: adipiscing elit...
74: </body>
75: </html>

```

Listing 5.20 Dynamisch aufklappendes Menü

Mit einer kleinen Veränderung im CSS-Code können wir noch eine Variante erzeugen. Wir ändern die Definition von `menu2` wie folgt:

```
#menu2 {
    position: relative;
    display: none;
    border-top: 1px solid #800000;
    background: #efefef;
    color: #000;
    font: bold 11px Verdana,Arial;
    text-decoration: none;
    width: 98px;
    z-index: 20;
}
```

Und erhalten dieses Verhalten:



Abbildung 5.30 Die zweite Menüebene ...



Abbildung 5.31 ... schiebt sich unter den ersten Menüpunkt.

Abwärtskompatibel mit Tricks

Die bisherigen Versionen funktionieren nur für Mozilla-Browser ab 1.0 und Opera 7. Die übrigen CSS-Browser (Internet Explorer 5 und 6, Opera 5 und 6) zeigen nur die erste Menüebene an. Damit sie auch noch die zweite Ebene darstellen, allerdings nicht dynamisch, müssen wir etwas tricksen.

Wir setzen die Angaben für alle Browser außer Mozilla zunächst fest und überschreiben sie dann mit den zur Dynamisierung nötigen Änderungen.

Im nächsten Kapitel zeige ich allerdings noch eine Möglichkeit, mit Hilfe von etwas JavaScript solche dynamischen Menüs auch auf Internet Explorern funktionsfähig zu machen.



Bleiben wir beim zweiten Beispiel. Die entscheidenden Angaben stecken in `#menu2`.

Es ist das `display: none`, das die Anzeige verhindert. Da eine dynamische Einblendung eben nur mit Mozilla und Opera 7 möglich ist, zeigen wir für alle anderen Browser die Ebene von Anfang an. Um zu zeigen, dass die Einträge der zweiten Ebene untergeordnet sind, setzen wir die Schriftdicke auch noch auf `normal`. Die im Folgenden fett markierten Anweisungen fallen also weg:

```
#menu2 {
    position: relative;
    display: none;
    border-top: 1px solid #800000;
    background: #efefef;
    color: #000;
    font: bold 11px Verdana,Arial;
    margin: 0 auto;
    padding: 0;
    text-decoration: none;
    width: 98px;
    z-index: 20;
}
```

Und nun setzen wir eine Anweisung ein, die nur Mozilla-Browser und Opera 7 beherrschen:

```
head:first-child+body #menu2 {
    display: none;
    font-weight: bold;
}
```

Diese etwas kryptische CSS-Anweisung markiert ein Element `#menu2`, welches sich innerhalb des Elements `body` befindet, was wiederum auf ein Element `head` folgt, das das erste Kind-Element eines anderen Elements ist. Das Ganze ist eigentlich nur umständlich ausgedrückt für:

```
#menu2 {
    display: none;
    font-weight: bold;
}
```

Siehe auch
Abschnitt 4.4.9

da `head:first-child` immer zutrifft – es gibt nur ein Element `<head>` in einem HTML-Dokument und das ist immer das erste Kind des Elements `<html>`. Nach `<head>` folgt auch immer `<body>`, sodass die Markierung `head:first-child+body` immer zutrifft. Aber da die meisten Browser `first-child` nicht interpretieren, führen sie die dazugehörigen Anweisungen nicht aus.

So erhalten wir ein Menü, das jeden Browser nach seinen Fähigkeiten behandelt (es funktioniert sogar mit Netscape 4, wenn Sie darauf verzichten, das Menü mit `float` zu positionieren).



Abbildung 5.32 Anzeige mit statischem Menü für Internet Explorer 6

Sie können diesen Trick auch für die erste Variante anwenden – Sie müssen nur darauf achten, dass alle Anweisungen, die nur die Anzeige für Mozilla-Browser und Opera 7 betreffen, in den zweiten Anweisungsblock ausgelagert werden.

5.8.2 CSS-Menüs mit Listen

Inzwischen hat sich bei den meisten Webentwicklern die Erkenntnis durchgesetzt, dass Menüs semantisch am korrektesten als Listen in HTML umgesetzt werden. Denn was ist ein Menü anderes als eine Liste von Links?

Nun sehen reine HTML-Listen alles andere als hübsch aus – glücklicherweise gibt uns CSS die Möglichkeit, Listen ganz nach unseren Vorstellungen zu gestalten.

Der HTML-Code einer solchen Liste ist einfach:

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

oder:

```
<ol>
<li><a href="..">Link 1</a></li>
<li><a href="..">Link 2</a></li>
<li><a href="..">Link 3</a></li>
</ol>
```

Vertikale Menüs

Vertikale Menüs sind recht einfach mit Listen zu gestalten. Sie haben im Grunde schon die richtige Form. Für ein einfaches Textmenü müssen wir nichts weiter ändern. Allerdings lassen sich per CSS Menüs auch erheblich aufwändiger gestalten – sodass die Verwendung des `<li>`-Tags nicht mehr zu erkennen ist.

Zunächst schalten wir einmal die normalen Aufzählungszeichen ab und neutralisieren die üblichen Abstände von HTML-Listen.

```
ul {
    list-style-type : none;
    margin          : 0;
    padding         : 0;
}
```

Dann setzen wir unsere eigenen Abstände und eine Hintergrundgrafik ein. Hintergrundgrafiken sind eine sehr flexible Methode, die üblichen Aufzählungszeichen zu ersetzen – wir können sie mit CSS beliebig relativ zum Text positionieren. Es lassen sich damit Designs realisieren, die mit normalen Aufzählungszeichen nicht möglich sind.

```
li {
    background-image : url(bullet.gif);
    background-repeat : no-repeat;
    background-position : 0 0.2em;
    padding-left      : 2em;
}
```

Wir können sogar rechtsbündige Listen erstellen:

```
ul {
    list-style-type : none;
    margin          : 0;
    padding         : 0;
    text-align      : right;
}
li {
    background-image : url(bullet-r.gif);
    background-repeat : no-repeat;
    background-position : 100% 0.2em;
    padding-right     : 2em;
}
```

Als Nächstes sorgen wir für ansprechende Rollover-Effekte (das können wir natürlich auch bei anderen Gelegenheiten anwenden):

```
li a {
    display      : block;
    color        : black;
    background-color : #f7941d;
    width        : 150px;
    padding      : 0.2em 0.6em;
    text-decoration : none;
}
```

und

```
li a:hover {
    background-color : #faddba;
}
```

Dies sieht doch fast schon wie ein grafisches Menü aus. Wenn wir jetzt noch ein paar Rahmen definieren, ist unser standardkonformes reines CSS-Menü nicht mehr von einem grafischen Menü mit JavaScript zu unterscheiden:

```
li a {
    display      : block;
    color        : black;
    background-color : #FBAF5D;
    width        : 150px;
    padding      : 0.2em 0.6em;
```

```

    text-decoration : none;
    border          : 1px solid #fff;
    border-bottom   : 1px solid #000;
    border-right    : 1px solid #000;
}

li a:hover {
    background-color : #faddba;
    border           : 1px solid #000;
    border-bottom    : 1px solid #fff;
    border-right     : 1px solid #fff;
}

```

Wir können die einzelnen Listenelemente auch mit Klassen markieren und so mit individuellen Hintergründen versehen. Das ist in zwei Situationen hilfreich:

1. Um ein Menü mit zusammenhängenden Hintergrundbildern mit RollOver-Effekt zu erstellen
2. Um die gerade ausgewählte Seite im Menü besonders zu markieren

Mit dem folgenden Code erstellen wir ein Menü, das mit einem durchgängigen Hintergrundbild versehen ist, welches als RollOver-Effekt abgedunkelt wird:

```

1:  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
    xhtml1-transitional.dtd">
2:  <html xmlns="http://www.w3.org/1999/xhtml">
3:  <head>
4:    <title>Listen als Menü</title>
5:    <meta http-equiv="Content-Type" content="text/html;
    charset=iso-8859-1" />
6:    <style type="text/css">
7:      /*<![CDATA[*]
8:        body {
9:          background-color: white;
10:        }
11:        ul {
12:          list-style-type : none;
13:          margin : 0;
14:          padding : 0;

```



```

15:     }
16:     li .a1 {
17:         margin: 0;
18:         background-image    : url(bullet_01.jpg);
19:         background-repeat   : no-repeat;
20:     }

```

Die Zeilen 16-20 werden noch sechsmal wiederholt – mit `.a2` - `.a7`.

Hier setzen wir den Normalzustand des Menüs fest. Die Menülinks werden später mit den Klassen `.a1` bis `.a7` versehen.

```

21:     li a {
22:         font-family: Georgia;
23:         font-style: italic;
24:         font-weight: bold;
25:         text-align: center;
26:         display: block;
27:         color: #000;
28:         width: 150px;
29:         height: 29px;
30:         margin: 0;
31:         text-decoration: none;
32:     }

```

Allgemeine Angaben, die für alle Links zutreffen:

```

33:     li .a1:hover {
34:         margin: 0;
35:         background-image    : url(bullet-o_01.jpg);
36:         background-repeat   : no-repeat;
37:         color: #fff;
38:     }

```

Hier wird der MouseOver-Zustand definiert. Das Hintergrundbild wird getauscht und die Schriftfarbe verändert – auch insgesamt siebenmal.

```

...
75:     /*]]>*/
76: </style>
77: </head>
78: <body>
79:     <ul>
80:         <li><a href="#" class="a1">Link 1</a></li>

```

```

81:      <li><a href="#" class="a2">Link 2</a></li>
82:      <li><a href="#" class="a3">Link 3</a></li>
83:      <li><a href="#" class="a4">Link 4</a></li>
84:      <li><a href="#" class="a5">Link 5</a></li>
85:      <li><a href="#" class="a6">Link 6</a></li>
86:      <li><a href="#" class="a7">Link 7</a></li>
87:  </ul>
88: </body>
89: </html>

```

Listing 5.21 Menü mit Hintergrund

Das sieht dann so aus:

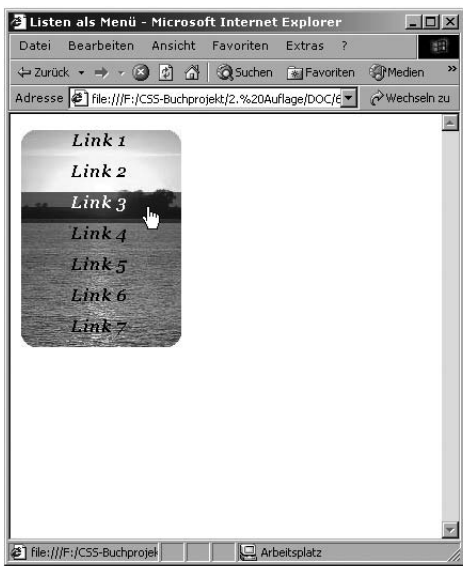


Abbildung 5.33 CSS-Menü mit geteilter Hintergrundgrafik

Eine weitere Variante ist das Kennzeichnen der aktuellen Seite im Menü. Dazu nehmen wir uns das Beispiel mit den Buttons noch einmal vor:

Anzeige der
aktuellen Seite
im Menü

```

li a {
    display      : block;
    color        : black;
    background-color : #FBAF5D;
    width        : 150px;
    padding      : 0.2em 0.6em;
}

```

```

        text-decoration : none;
        border          : 1px solid #fff;
        border-bottom   : 1px solid #000;
        border-right    : 1px solid #000;
    }

    li a:hover {
        background-color : #faddba;
        border           : 1px solid #000;
        border-bottom    : 1px solid #fff;
        border-right     : 1px solid #fff;
    }

```

Hinzu kommt eine Klassendeklaration:

```

li a.aktuell {
    display          : block;
    color            : black;
    background-color : #0b256b;
    width            : 150px;
    padding          : 0.2em 0.6em;
    text-decoration  : none;
    border           : 1px solid #fff;
    border-bottom    : 1px solid #000;
    border-right     : 1px solid #000;
}

```

Nun müssen wir nur auf der jeweils aktuellen Seite den entsprechenden Eintrag (genauer: das `<a href="..">`) mit der Klasse `.aktuell` versehen.

Wir können es auch umgekehrt machen. Wenn wir die Technik anwenden, die im Beispiel Lycos erklärt ist (Abschnitt 6.2 – Verwendung einer ID im `<body>`) und jeder Seite (oder jeder Sektion) eine eigene ID zuweisen, können wir den aktuellen Menüpunkt über kombinierte Selektoren ansprechen.

Wenn wir drei Seiten mit den folgenden `body`-Tags haben:

```

<body id="start">
<body id="inhalt">
<body id="kontakt">

```

und ein solches Menü:

```

<ul>
  <li><a href="#" class="a1">Link 1</a></li>
  <li><a href="#" class="a2">Link 2</a></li>
  <li><a href="#" class="a3">Link 3</a></li>
</ul>

```

können wir mit einer einzigen Anweisung

```

#start .a1, #inhalt .a2, #kontakt .a3 {
  font-weight: bold;
}

```

dafür sorgen, dass auf der aktuellen Seite der passende Menüpunkt fett gesetzt wird. Das lässt sich natürlich auf Bereiche einer Website anwenden.

Mit der Listenfunktion lassen sich auch Submenüs semantisch korrekt darstellen: als verschachtelte Listen.

**Verschachtelte
Menüs als Listen**

```

<ul>
<li><a href="..">Link 1</a></li>
<li><a href="..">Link 2</a><br />
  <ul>
    <li><a href="..">Sublink A</a></li>
    <li><a href="..">Sublink B</a></li>
  </ul>
</li>
<li><a href="..">Link 3</a></li>
</ul>

```

Die CSS-Definition sieht zunächst aus wie gehabt:

```

ul {
  list-style-type : none;
  margin          : 0;
  padding         : 0;
}
li {
  background-image : url(bullet.gif);
  background-repeat : no-repeat;
  background-position : 0 0.2em;
  padding-left      : 2em;
}

```

Es folgt hier aber eine weitere Definition für die zweite Ebene. Dazu nutzen wir kombinierte Selektoren: `li li` bedeutet einfach »ein Listenelement, welches sich in einem anderen Listenelement befindet«. Dieses Spiel können Sie beliebig fortsetzen, um tiefer verschachtelte Menüs zu erzeugen.

```
li li {
    background-image    : url(bullet.gif);
    background-repeat    : no-repeat;
    background-position : 0 1.2em;
    padding-left         : 3em;
}
```

Horizontale Menüs

Sie können aber mit Listen noch viel mehr erreichen: Auch horizontale Menüs sind kein Problem. Der Trick besteht darin, die eigentlich untereinander positionierten `<li>`-Elemente mittels `float` nebeneinander zu setzen.



```
1:  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
    xhtml1-transitional.dtd">
2:  <html xmlns="http://www.w3.org/1999/xhtml">
3:  <head>
4:  <title>Unbenanntes Dokument</title>
5:  <meta http-equiv="Content-Type" content="text/html;
    charset=iso-8859-1" />
6:  <style type="text/css">
7:  <!--
8:  ul {
9:      list-style-type : none;
10:     margin           : 0;
11:     padding          : 0;
12:  }
13:  li {
14:     width            : 100px;
15:     margin           : 5px;
16:     float            : left;
17:     text-align       : center;
18:  }
19:
```

```

20: li a {
21:     text-decoration : none;
22:     color            : black;
23:     display          : block;
24:     width            : 100%;
25:     background-color : #dedede;
26:     padding          : 2px 0;
27:     border           : 1px solid #000;
28: }

```

Das `display: block` ist hier notwendig, damit sich die Links (normalerweise eingebundene Elemente) auf die gesamte Fläche der `<li>`-Elemente ausdehnen lassen (mit `width: 100%` für den Internet Explorer) und somit der ganze »Button« anklickbar ist.

```

29: li a:hover {
30:     background-color : #efefef;
31: }
32:
33: -->
34: </style>
35: </head>
36:
37: <body>
38: <ul>
39:   <li><a href="..">Link 1</a></li>
40:   <li><a href="..">Link 2</a></li>
41:   <li><a href="..">Link 3</a></li>
42: </ul>
43: </body>
44: </html>

```

Listing 5.22 Horizontales CSS-Menü mit Liste

Auf der Website List-O-Matic ist der umgekehrte Weg beschrieben: http://www.maxdesign.com.au/presentation/listutorial/horizontal_introduction.htm (Linkcode 0059).



Mit einem

```
ul li { display: inline }
```

werden die Block-Elemente `<ul>` und `<li>` zu eingebundenen Elementen umdefiniert, die dann frei hintereinander fließen.

Dort sind auch viele weitere Ressourcen zu CSS-Listen zusammengetragen – ein Besuch lohnt schon wegen der List-O-Matic.

CSS-Flyout-Menüs

Mit den beschriebenen Techniken lässt sich auch ein Flyout-Menü »CSS powered« (zumindest in standardkonformen Browsern) erzeugen.

Wir beginnen mit einer simplen verschachtelten Liste:

```
<ul>
  <li><a href="#">Link 1</a></li>
  <li><a href="#" class="aktuell">Link 2</a></li>
  <li><a href="#">Link 3</a>
    <ul>
      <li><a href="#">Link 3.1</a></li>
      <li><a href="#">Link 3.2</a></li>
    </ul></li>
  <li><a href="#">Link 4</a></li>
  <li><a href="#">Link 5</a></li>
</ul>
```

Diese Liste soll zu einem Menü werden, in dem die zweite Ebene seitlich ausklappt:

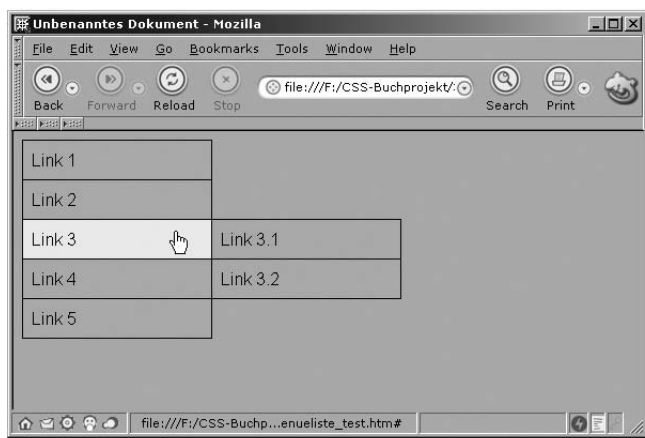


Abbildung 5.34 Listen ausgeklappt

Sehen wir uns den Code einmal genauer an:

```

1: <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
   Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
   xhtml1-transitional.dtd">
2: <html xmlns="http://www.w3.org/1999/xhtml">
3: <head>
4:   <title>CSS-Flyout-Menü</title>
5:   <meta http-equiv="Content-Type" content="text/html;
   charset=iso-8859-1" />
6:   <script type="text/javascript">
7:     <!--//--><![CDATA[//><!--
8:       ieHover = function() {
9:         var LIs = document.getElementsByTagName("LI");
10:        for (var i=0; i<LIs.length; i++) {
11:          LIs[i].onmouseover=function() {
12:            this.className+=" iehover";
13:          }
14:          LIs[i].onmouseout=function() {
15:            this.className=this.className.replace
16:              (new RegExp(" iehover\\b"), "");
17:          }
18:        }
19:      }
20:      if (window.attachEvent) window.
        attachEvent("onload", ieHover);
21:    //--><![]]>
22:   </script>

```

Hier kommt gleich der spannendste Teil: ein kleines JavaScript, mit dem Sie für den Internet Explorer das fehlende `:hover` nachprogrammieren. Das Skript sucht aus dem aktuellen Dokument alle `<li>`-Elemente heraus (Zeile 9). Danach weist es in einer Schleife immer dann, wenn das JavaScript-Ereignis »onmouseover« eintritt (also jemand mit der Maus auf das Element fährt), dem betreffenden Element eine zusätzliche Klasse »iehover« zu. Umgekehrt wird die Klasse entfernt, wenn »onmouseout« auftritt. Ich habe das Skript hier zum besseren Verständnis direkt in die Seite geschrieben. Wenn Sie es einsetzen, sollten Sie es in eine eigne Datei auslagern. Das Skript wurde von *Patrick Griffiths* und *Dan Webb* entwickelt und unter dem Titel »Suckerfish Dropdown« bei **alistapart.com** vorgestellt (nein, ich weiß auch nicht, woher der Name kommt).

Suckerfish
Dropdown

```

23: <style type="text/css">
24: /*<!CDATA[*]
25:     body {
26:         background-color: silver;
27:     }
28:     ul {
29:         margin: 0;
30:         padding: 0;
31:         list-style: none;
32:         width : 11em;
33:     }
34:     li {
35:         position : relative;
36:         float : left;
37:         line-height : 1.25em;
38:         margin-bottom : -1px;
39:         width: 11em;
40:         background-color: #fbaf5b;
41:     }
42:     li a {
43:         display: block;
44:         padding : 0.5em;
45:         width: 11em;
46:         w\idth : 10em; /* Boxmodelhack für IE */
47:         color : black;
48:         text-decoration : none;
49:         border : 1px solid black;
50:     }
51:     li a:hover {
52:         background-color : #faeedc;
53:     }

```

Hier definieren wir die erste Ebene unseres Menüs – analog zu den Beispielen der vorgehenden Seiten. Die `float`-Anweisung benötigen wir, um einen häßlichen Spalt beim Internet Explorer zu vermeiden; mit `margin-bottom: -1` überlagern wir die Rahmenlinien aufeinander folgender Kästen, um doppelte Linien zu vermeiden. Wir müssen eine explizite Zeilenhöhe angeben, damit wir nachher die nächste Ebene positionieren können.

```

54:      li ul {
55:          position : absolute;
56:          left: -999em;
57:          margin-left : 11.05em;
58:          margin-top : -2.35em;
59:      }

```

Jetzt kommt die zweite Ebene, die wir mit `left: -999em` nach links aus dem Bild verschwinden lassen. Den gleichen Effekt könnten Sie auch mit `display: none` erreichen. Dies führt aber in Screenreadern zu Problemen, da sie den solcherart verborgenen Bereich auch nicht mehr vorlesen.

```

60:      li:hover ul, li.iehover ul {
61:          left: auto;
62:      }

```

Mit dieser Anweisung holen wir das Menü der zweiten Ebene wieder hervor, wenn der jeweilige Punkt der ersten Ebene mit der Maus ausgewählt wird. Hier erkennen Sie die Klasse `.iehover` wieder, die im JavaScript zugewiesen wurde. Der Trick besteht also darin, dem Internet Explorer statt `.hover` eine gleich formatierte Klasse mit JavaScript dynamisch zuzuweisen. Dieses Verfahren ist natürlich nicht nur auf `.hover` beschränkt – Sie können damit einige Lücken des IE bei der Behandlung von Selektoren ausbügeln.

```

63:      /*]]*//
64:      </style>
65: </head>
66: <body>
67:     <ul>
68:         <li><a href="#">Link 1</a></li>
69:         <li><a href="#" class="aktuell">Link 2</a></li>
70:         <li><a href="#">Link 3</a>
71:         <ul>
72:             <li><a href="#">Link 3.1</a></li>
73:             <li><a href="#">Link 3.2</a></li>
74:         </ul>
75:     </li>
76:     <li><a href="#">Link 4</a></li>
77:     <li><a href="#">Link 5</a></li>
78: </ul>

```

```
79: </body>
80: </html>
```

Listing 5.23 Listing: Ein browserübergreifendes CSS-Flyout-Menü mit Sicherheitsnetz für Internet Explorer

Diese Lösung funktioniert mit allen Gecko-Browsern (Mozilla, Firefox), Opera ab 7.54, Safari und den Internet Explorern 5.5 und 6.



Der Entwickler Dean Edwards hat aus einer ganzen Reihe solcher Tricks eine Erweiterung zusammengestellt, die er »IE7« nennt. Damit können Sie durch Einbindung eines kleinen JavaScripts alle Internet Explorer, die auf Ihre Site kommen, temporär zum »Internet Explorer 7« hochrüsten.

Eine beeindruckende Arbeit, die Sie unter **<http://dean.edwards.name/ie7/>** (Linkcode 0007) bewundern und herunterladen können.

List-O-Matic und Listamatic

List-O-Matic ist ein Online-Generator: Sie können sich dort eigene CSS-Menüs aus Listen erstellen lassen. Sie müssen lediglich Ihre Menütex te eingeben und aus zehn verschiedenen Designs wählen. Einfacher geht es nicht mehr! Die fertigen CSS-Codes lassen sich natürlich nach eigenen Vorstellungen weiter anpassen und verändern.

Die List-O-Matic finden Sie unter:

<http://www.accessify.com/tools-and-wizards/list-o-matic/list-o-matic.asp> (Linkcode 0060)

Auch die Website *Listamatic* befasst sich mit Listen. Hier finden Sie eine Vielzahl von CSS-gesteuerten Listen, die Sie nach Belieben einsetzen und anpassen können. Die einzelnen Beispiele sind ausführlich erläutert (in englischer Sprache).

<http://css.maxdesign.com.au/listamatic/> (Linkcode 0061) und

<http://css.maxdesign.com.au/listamatic2/> (Linkcode 0062)

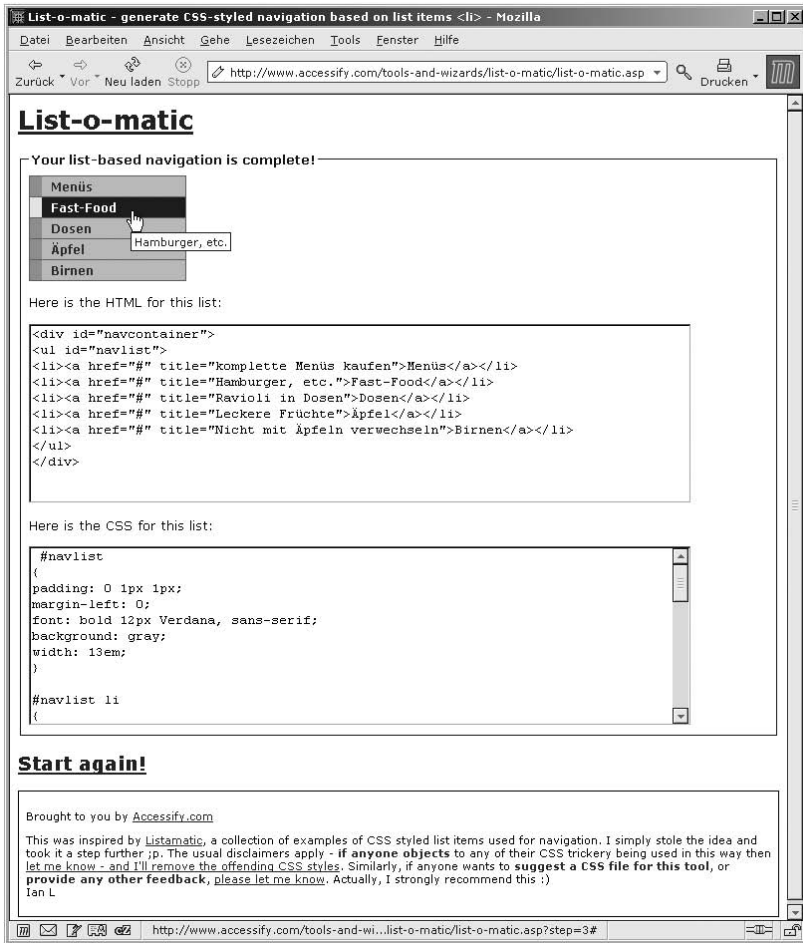


Abbildung 5.35 Mit List-O-Matic zum eigenen CSS-Menü

List-O-Rama

Auf der Dreamweaver-Community DMX-Zone bietet der Entwickler und Gründer George Petrov eine nützliche Erweiterung für Dreamweaver an, mit der sich automatisch CSS-Menüs auf Basis des `<li>`-Elements (beschrieben in Abschnitt 5.8.2) erstellen lassen.

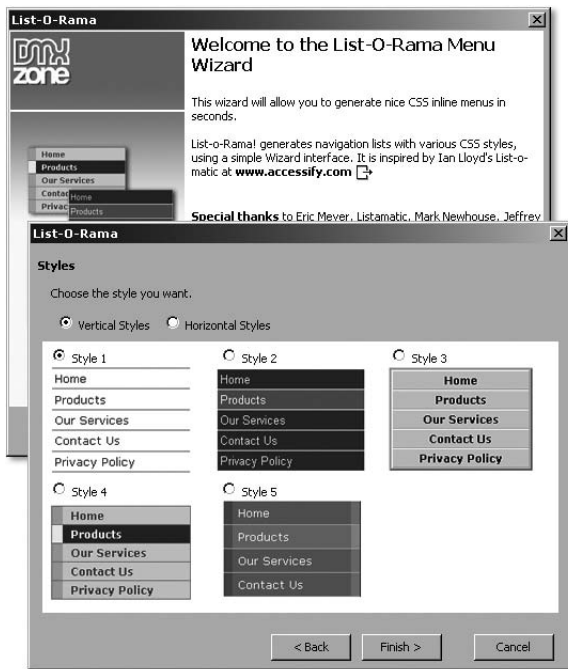


Abbildung 5.36 CSS-Menüs mit List-O-Rama

Sie müssen die Dreamweaver-Extension nur installieren und können ab sofort in der Befehlssymbolsite (❶ in Abbildung 7.9) ein CSS-Menü in Ihre Seiten integrieren.

Sie finden die Extension in der DMX-Zone unter <http://www.dmx-zone.com> (Linkcode 0063) (ein Besuch lohnt sich ohnehin).

5.9 Bildergalerie

Das Layout von Bildergalerien wird typischerweise mit Tabellen erstellt. Dabei werden Thumbnails (kleine Vorschaubilder) in einem Gitter angezeigt. Die Bilder sind mit Unterschriften versehen und auf die Fotos in Originalgröße verlinkt.

Bildergalerie
mit CSS

Im »traditionellen« Tabellenlayout würden wir jedes Bild in eine Tabellenzelle setzen und jeder Bildunterschrift ebenfalls eine Tabellenzelle spendieren. Jetzt können wir dieses Layout ohne Tabellen nur durch CSS realisieren (es ist sogar besser, weil es sich automatisch an die Seitenbreite anpasst).

Sie finden den vollständigen Quellcode und ein Anwendungsbeispiel auf der Buch-CD.



5.12.4 JavaScript mit Sicherheitsnetz

Sowohl serverseitige als auch clientseitige Lösungen haben ihre Vor- und Nachteile. Was liegt also näher, als die beiden Ansätze miteinander zu verbinden?

Mit freundlicher Hilfe von David Andersson (aka Liorean) habe ich ein Kombinationsskript aus PHP und JavaScript erstellt (der gesamte JavaScript-Part stammt von Liorean), das die besten Eigenschaften beider Welten vereinigt. Bei diesem Skript (eigentlich sind es ja mehrere) wird JavaScript genutzt, wenn es aktiviert ist. Ansonsten werden die Stylesheets mit PHP umgeschaltet. Ein Cookie speichert die Auswahl, sodass Besucher bei den nächsten Besuchen ihre alte Auswahl wieder zugewiesen bekommen.

Das Beste aus beiden Welten

Der Styleswitcher ermöglicht es, aus einer gegebenen Auswahl von Stylesheets eines durch den Website-Besucher auswählen zu lassen und auf die aktuelle Site anzuwenden. Die Auswahl wird durch einen Cookie gespeichert und bleibt beim Wechseln der Seiten erhalten. Auch bei späteren Besuchen wird das einmal gewählte Stylesheet wieder verwendet.

Der Styleswitcher funktioniert standardmäßig mit JavaScript und ist in dieser Form auch kompatibel zur Stylesheetauswahl in Mozilla/Gecko-Browsern über das Ansicht-Menü. Falls JavaScript beim Benutzer nicht aktiviert ist, wird die Stylesheet-Umschaltung über ein PHP-Skript realisiert (dazu ist natürlich ein Server mit PHP-Unterstützung erforderlich).

styleswitcher.php	Anwendungsbeispiel: Die Datei zeigt, wie das JavaScript und das PHP-Skript eingebunden werden und die darin enthaltenen Funktionen angesprochen werden können.
themeswitch-ce.js	JavaScript-Funktionen: Alle JavaScript-Funktionen. Gegenüber dem Originalskript von Liorean ist nur die Zeile 42 verändert. Dort wird die aufrufende Domain beim Schreiben des Cookies übergeben (und die Gültigkeit des Cookies auf 365 Tage gesetzt).
themeswitch-php.php	PHP-Funktionen: Hier sind alle Funktionen für die Funktion ohne JavaScript untergebracht.

Bestandteile

Standard.css, Serioes.css, Wild.css	Beispiel-Stylesheets: Diese Stylesheets werden im Anwendungsbeispiel verwendet.
themeswitch.html	Original-Anleitung zu Lioreans Skript (engl.)
themeswitch.zip	Original-Skripte von Liorean: Die Original-Skripte enthalten nur die JavaScript-Funktionalität. Interessant zum Studium oder für eigene Modifikationen

Anpassung **styleswitcher.php**

In alle HTML-Dateien, die den Styleswitcher verwenden, müssen die umschaltbaren Stylesheets und die Links zum Umschalten eingebunden werden.

Im mitgelieferten Anwendungsbeispiel sieht das so aus:

In den Zeilen 12-14 werden die Stylesheets angegeben. Es können natürlich auch mehr oder weniger als drei sein. Das Standard-Stylesheet wird mit der Angabe `rel="stylesheet"`, alle anderen werden mit `rel="alternate stylesheet"` eingebunden.

In der Zeile 20 wird das Stylesheet angegeben, welches verwendet werden soll, wenn der Besucher zum ersten Mal auf die Site kommt. Im Anwendungsbeispiel ist es *Standard*.

In den Zeilen 24-29 werden die Links zum Umschalten der Stylesheets gezeigt. Hier müssen Sie Ihre Stylesheetnamen einsetzen. Achten Sie darauf, dass auch die Titel der Links mit den Dateinamen übereinstimmen (ohne die Endung.css).

themeswitch-ce.js

In der Zeile 42 ersetzen Sie `css-praxis.de` durch die Domain, auf der das Skript zum Einsatz kommt.

themeswitch-php.php

In Zeile 2 tragen Sie alle verwendeten Stylesheets in das Array ein (eine Sicherheitsfunktion, die das Ausführen von potenziell schädlichem Code verhindern soll).

In Zeile 7 steht der Name des Standard-Stylesheets (passenderweise heißt es auch Standard ...).

In Zeile 10 ersetzen Sie wieder den Domainnamen.



Sie finden die kompletten Skripte inklusive einer Einbindungsanleitung und Beispielen auf der Buch-CD-ROM.

8.2 Pseudo-Elemente und Pseudo-Formate

Pseudo-Elemente und -Formate sind besondere Elemente, die im HTML-Dokument nicht notiert werden, sondern spezielle Eigenschaften oder Zustände eines anderen Elements beschreiben, z.B. die erste Zeile eines Absatzes (siehe auch Abschnitt 2.1.8).

8.2.1 :link

Beschreibung

Mit der Pseudo-Klasse `:link` werden Hyperlinks (Anker mit Verweisziel) angesprochen, so wie sie auch im `<body>`-Tag eines HTML-Dokuments mit dem Attribut `link` definiert werden können. Im Gegensatz dazu kann per CSS allerdings nicht nur die Farbe des Links geändert werden.

Standardwert

Keiner

Vererbung

Ja

Syntax-Beispiele

```
:link {  
    color: red;  
    font-size: 1.2em  
}  
a:link {  
    color: red;  
    font-size: 1.2em  
}  
a.second:link {  
    color: red;  
    font-size: 1.2em  
}
```

Kompatibilität

	NN4	NN7	Mz 1.4	Mz 1.8/FF	IE4	IE5	IE6	O5	O6	O7	Sf 1.2	Konq
PC	j ¹	j	j	j	j	j	j	j	j	j	-	-
Mac/Linux	j ¹	j	j	j	j	j	-	j	j	-	j	j

1 Der Netscape Navigator 4.x überschreibt die Pseudo-Klassenselektoren immer durch die Angaben für das Element `a`. Eigentlich müsste `a:link` immer die Anweisungen für `a` überschreiben, da es eine höhere Spezifität besitzt (siehe auch Abschnitt 2.3, *Rangfolge und Kaskadierung*).

Lösung: Definieren Sie das Element `a` nicht, wenn Sie für die Pseudo-Elemente eigene Anweisungen notieren wollen.

Bemerkungen

Sie können durch die Kombination von `:link` mit einer Klasse auch mehrere verschiedene Link-Formatierungen festlegen:

Dabei sind die Änderungen natürlich nicht auf die Linkfarbe beschränkt – Schriftart, -größe oder Hintergrundfarbe sind nur einige Möglichkeiten. Beliebt ist auch das Abschalten der Unterstreichung durch `a:link {text-decoration: none}`.



Dazu ein Beispiel: Auf einer Webseite befinden sich zwei farbige Bereiche – die Farbe des einen ist der Linkfarbe so ähnlich, dass »normale« Links dort nicht sichtbar wären. Also wollen wir die Linkfarbe für diesen Abschnitt ändern. Wir verwenden dazu eine eigene Klasse für den betreffenden Bereich und einen kombinierten Selektor (siehe Abschnitt 2.1.2). So sieht der Code aus:

```
1:  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
    xhtml1-transitional.dtd">
2:  <html xmlns="http://www.w3.org/1999/xhtml">
3:  <head>
4:    <title>Unterschiedliche Links</title>
5:    <meta http-equiv="Content-Type" content="text/
      html; charset=iso-8859-1" />
6:    <style type="text/css">
7:      /*<![CDATA[*]
8:        a:link {
9:          color: #003399;
10:        }
11:        .linkbereich2 {
```

```

12:         background-color: #003399;
13:         color: #dedede;
14:     }
15:     .linkbereich2 a:link {
16:         color: #ccff99;
17:         font-weight: bold;
18:     }
19:     /*]]>*/
20: </style>
21: </head>
22: <body>
23:     <p>Dies ist ein Dokument mit unterschiedlichen
        Links.
24:     In diesem ersten Abschnitt werden die <a href="#">
        Links ganz normal</a> formatiert: blau mit Unter-
        streichung.
25:     </p>
26:     <div class="linkbereich2">
27:         <p>&nbsp;</p>
28:         <p>In einem zweiten Abschnitt ist die Hinter-
            grundfarbe so, dass die <a href="#">Links
29:         </a> nicht sichtbar wären - darum
30:         <a href="#">färben</a> wir sie um.</p>
31:         <p>&nbsp;</p>
32:     </div>
33: </body>
34: </html>

```

Listing 8.1 Unterschiedliche Links

Das sieht dann folgendermaßen aus:

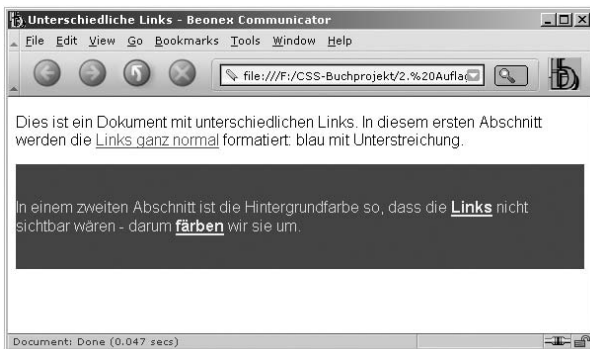


Abbildung 8.1 Unterschiedlich formatierte Links

Bei allen Möglichkeiten, Links zu formatieren – auch in den folgenden Abschnitten –, sollten Sie nicht vergessen, dass Links einen wesentlichen Teil der Bedienelemente einer Website ausmachen und als solche klar erkennbar sein müssen. Die im Web üblichen Kennzeichen von Links sind ihre blaue Farbe und die Unterstreichung. Wenn Sie beides ändern – woher soll der Besucher wissen, was ein Link ist?

8.2.2 :visited

Beschreibung

:visited markiert bereits besuchte Hyperlinks und entspricht damit dem HTML-Attribut vlink von <body>.

Standardwert

Keiner

Vererbung

Ja

Syntax-Beispiele

```
a:visited {
    color: red;
    font-size: 1.2em
}
a.hauptmenue:visited {
    color: red;
    font-size: 1.2em
}
```

Kompatibilität

	NN4	NN7	Mz 1.4	Mz 1.8/FF	IE4	IE5	IE6	O5	O6	O7	Sf 1.2	Konq
PC	j	j	j	j	j	j	j	j	j	j	–	–
Mac/Linux	n	j	j	j	j	j	–	j	j	–	j	j

Bemerkungen

a:visited muss im Stylesheet nach a:link definiert werden, damit die :visited-Anweisungen nach den Regeln der Kaskadierung (siehe Abschnitt 2.3, *Rangfolge und Kaskadierung*) die :link-Anweisungen überschreiben.



8.2.3 :hover (CSS2)

Beschreibung

Mit `:hover` wird der Zustand markiert, wenn der Mauszeiger über einem aktiven Element schwebt, ohne bereits gedrückt zu sein.

Standardwert

Keiner

Vererbung

Ja

Syntax-Beispiele

```
a:hover {
    color: red;
    font-size: 1.2em
}
a.hauptmenue:hover {
    color: red;
    font-size: 1.2em
}
```

Kompatibilität

	NN4	NN7	Mz 1.4	Mz 1.8/FF	IE4	IE5	IE6	O5	O6	O7	Sf 1.2	Konq
PC	n	j	j	j	j	j	j	j	j	j	–	–
Mac/Linux	n	j	j	j	j	j	–	j	j	–	j	j

Alle Browser außer Mozilla unterstützen `:hover` nur für Links. Bei Mozilla können Sie auch andere Elemente verändern, z.B. `<p> per :hover.`

Bemerkungen

`a:hover` muss im Stylesheet nach `a:link` und `a:visited` definiert werden, damit die `:hover`-Anweisungen nach den Regeln der Kaskadierung (siehe Abschnitt 2.3, *Rangfolge und Kaskadierung*) die `:link`-Anweisungen bzw. die `:visited`-Anweisungen überschreiben.



8.2.4 :focus (CSS2)

Beschreibung

`:focus` markiert ein Element, das vorausgewählt ist, d.h. mit der Taste  aktiviert werden könnte. Sie können beispielsweise mit der -Taste durch die Links einer Webseite navigieren. Vorausgewählte Elemente werden von manchen Browsern durch eine dünne Linie um das Element gekennzeichnet.

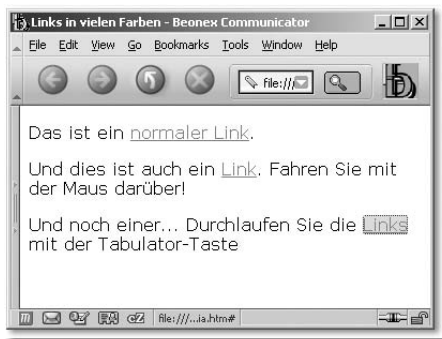


Abbildung 8.2 Ein vorausgewählter Link mit hellgrauer Hintergrundfarbe (siehe Syntax-Beispiele)

Standardwert

Keiner

Vererbung

Ja

Syntax-Beispiele

```
a:focus {  
    color: red;  
    background-color: #daded;  
}  
  
a.hauptmenue:focus {  
    color: red;  
    font-size: 1.2em;  
}
```

Kompatibilität

	NN4	NN7	Mz 1.4	Mz 1.8/FF	IE4	IE5	IE6	O5	O6	O7	Sf 1.2	Konq
PC	n	j	j	j	n	n	n	n	n	n ²	–	–
Mac/Linux	n	j	j	j	n	t ¹	–	n	n	–	j	j

1 Der Internet Explorer 5 für Macintosh unterstützt `:focus` nicht für alle Eigenschaften – so sind zwar Änderungen der Hintergrundfarbe, aber keine Größenänderungen von Elementen möglich.

2 Obwohl alle mir zugänglichen Dokumentationen behaupten, Opera 7 beherrsche `:focus`, konnte ich das nicht feststellen ...

Bemerkungen

Pseudo-Klassen für Links können auch kombiniert werden. So markiert `a:focus: hover` ein vorausgewähltes Element, über dem der Mauszeiger schwebt.

`:focus` ist die Entsprechung von `:hover` für die Tastaturbenutzung und sollte für alle mit `:hover` ausgezeichneten Elemente zusätzlich angegeben werden. Auch wenn `:focus` momentan nur von Mozilla-Browsern unterstützt wird, so schadet es sicher nicht, eine zusätzliche Hilfe für Tastaturbenutzer anzugeben. Im Zusammenhang mit Formularfeldern ergeben sich interessante Möglichkeiten, die ich an einem Beispiel in Abschnitt 5.10, *Schönere Formulare*, zeige.



8.2.5 `:active`

Beschreibung

`:active` markiert den Hyperlink, der gerade aktiviert ist (»gedrückt wird«) und entspricht damit dem HTML-Attribut `alink`.

Standardwert

Keiner

Vererbung

Ja

Syntax-Beispiele

```
a:active {
    color: red;
```

```

    font-size: 1.2em;
}
a.hauptmenue:active {
    color: red;
    font-size: 1.2em;
}

```

Kompatibilität

	NN4	NN7	Mz 1.4	Mz 1.8/FF	IE4	IE5	IE6	O5	O6	O7	Sf 1.2	Konq
PC	f	j	j	j	j	j	j	j	j	j	–	–
Mac/Linux	f	j	j	j	j	j	–	j	j	–	j	j

Bemerkungen



`:active` muss im Stylesheet nach `:hover` (und nach `:link` und `:visited` und `:focus`) definiert werden, damit die `:active`-Anweisung nach den Regeln der Kaskadierung (siehe Abschnitt 2.3, *Rangfolge und Kaskadierung*) die vorangegangenen Anweisungen überschreibt.

Damit ergibt sich eine eindeutige Reihenfolge, in der Anweisungen zur Formatierung von Links notiert werden müssen.

1. `a`: (Falls aus irgendeinem Grund Anweisungen für das `<a>`-Element festgelegt werden sollen. Im Allgemeinen lässt man es einfach weg – schon um keine Probleme mit dem oben erwähnten Netscape-4-Bug zu bekommen.)
2. `link`
3. `visited`
4. `focus`
5. `hover`
6. `active`

8.2.6 :lang (CSS 2)

Beschreibung

Mit `:lang` ist es möglich, ein Element aufgrund der ihm zugewiesenen natürlichen Sprache anzusprechen.

Index

Symbols

!important 60
(399, 400, 402, 408, 409, 411, 524
@font-face 431
@import-Weiche 118

A

A List Apart 261
abbr 288
Absolute Einheiten 75
Abstand → margin 463
Abwärtskompatibilität 152, 218
Accessibility 150, 153, 381
acronym 288
aDesigner 151
Adobe 372
Aktion Mensch 273
alt 381
alternative Stylesheets 258, 284
Alt-Finder 380
Anführungszeichen 410, 412
Anzeigemodus → display 505
ASP 265, 367
Asterisk 380
Attribut-Selektor 51, 380
Aufzählungszeichen 213
Aufzählungszeichen → list-style-image 503
Aufzählungszeichen → list-style-type 501
Ausrichtung 445
Außenabstand 73
Äußerer Abstand → margin 73, 463
Äußerer Rand 73
azimuth 537

B

background 461
background-attachment 459
background-color 455
background-image 456
background-position 460
background-repeat 457
behindertengerechtes Webdesign 163
Benutzerauswahl 261

Benutzeroberfläche 515
Bildergalerie 240
Bildschirm-Lineal 388
Bildunterschrift 413
BITV 150
blinde GIFs 94
Blindows 95
Block-Element 44, 74, 506
block-level elements 74
Bobby 153
Bookmarklets 377
border 73, 473
 border-bottom 473
 border-left 473
 border-right 473
 border-top 473
border-bottom → border 473
border-bottom-color → border-color 468
border-bottom-style → border-style 469
border-bottom-width → border-width 467
border-collapse 511
border-color 468
border-left → border 473
border-left-color → border-color 468
border-left-style → border-style 469
border-left-width → border-width 467
border-right → border 473
border-right-color → border-color 468
border-right-style → border-style 469
border-right-width → border-width 467
border-spacing 512
border-style 469
border-top → border 473
border-top-color → border-color 468
border-top-style → border-style 469
border-top-width → border-width 467
border-width 467
bottom 73, 494
Box Acid Test 101

- Box Model 73
- Box Model Hack 120, 121
- box-sizing 491
- Braille-Zeilen 94
- Breadcrumbs 287
- Breite → width 73, 474
- Browser 40
- Browseranteile 111
- Browserhacks 116
- Browserkrieg 276
- Browser-Sniffer 129
- Browser-Statistik 112
- Browserweiche 116, 269
- Bundesbehindertengleichstellungs-
gesetz 150
- Buttons 247

C

- Calipers 388
- caption-side 510
- Cascade 362
- Cascading Stylesheets 25
- CDATA 88
- CD-ROM 541
- charakter data → CDATA 88
- child 49
- CLASS 47
- Class-Selektoren 47
- clear 483
- clip 478
- ColdFusion 367
- color 454
- Conditional Comments 128, 157
- content 412
- Cookie 262, 263
- counter () 413
- counter-increment 416
- counter-reset 417
- counters() → counter () 413
- CSS Zen Garden 143
- CSS-Checker 358
- CSS-Editoren 357
- CSS-Frames 208
- CSS-Link 266
- CSS-Menüs 217
- css-validator 153
- cue 532
- cursive → kursive Schriften 421

- cursor 515

D

- Definitionsliste 294, 298
- Deklaration 28
- Developer-Exchange 367
- DHTML 550
- DIN A4 522
- direction 452
- display 505
- Doctype 112
- Doctype → Dokumententyp 86
- Doctype-Switching 112
- Document Objekt Model 548
- Dokumententyp 86
- DOM → Document Objekt Model
548
- Dreamweaver 89, 278, 366
- Dreispartiges Layout 180
- Drucken 256, 284
- Druck-Style Sheet 303
- Druckversion 254, 256, 257
- Dynamische Menüs 213, 218
- dynamische Webseiten 548

E

- Eigenschaft 28
- einfach-fuer-alle.de 273
- Eingabefelder 247
- eingebundene Elemente → inline
elements 74, 506
- Elementselektoren 43
- em 162, 163
- emacspeak 94
- empty-cells 513
- Entwurfsverfahren 144
- ersetzte Elemente → replaced
elements 74
- ex 76
- Exchange 367
- expressions 210
- eXtensible Markup Language 90
- eXtensible Stylesheets 90
- Extensions 367
- externe Ressourcen 74

F

- Fahrner Image Replacement 148, 287

- Farben 77, 454
 - Dezimalwerte 77
 - Hexadezimal 77
 - Prozentwerte 77
 - Schlüsselwörter 77
- Farbwerte 74
- Fehlersuche 145
- Feststehende Bereiche 201
- Feststehendes Menü 201
- fieldset 301
- filter 520
- Firefox 104, 390
- Flexibler Dreispalter 188
- float 292, 479
- Flyout-Menüs 234
- Folgeelement-Selektoren 51
- font 155, 380, 429
- font-family 420
- Font-Finder 380
- font-size 426
- font-size-adjust 427
- Fontsizer 163
- font-stretch 429
- font-style 423
- font-variant 424
- font-weight 425
- Formulare 244, 246, 301
- Formularelemente 244
- Frames 202
- Frames → CSS-Frames
- fremdsprachlich 294

G

- gleichlange Spalten 192
- GoLive 372
- Grundlinie 446

H

- Handheld 70
- height 73, 475
 - max-height 475
 - min-height 475
- Hintergrundfarbe → background-color 455
- Hintergrundgrafik → background-image 456
- Hintergrundmusik → play-during 533
- Hochformat 522

- Hochpass-Filter 124
- Höhe → height 73, 475
- Homepage Reader 94, 151
- Horizontale Menüs 232
- hr 297
- href 255
- HTML 19, 85
- HTML-Formular 246

I

- ID 47
- ID-Selektoren 47
- IE/Mac-Filter 127
- IE7 85
- Important-Anweisung 64
- Importierte Stylesheets 69
- Inch → Zoll 75
- Informationsarchitektur 278
- inheritance 57
- Initiale 406, 549
- inline elements 74
- inline styles 65
- Innenabstand 73
- Innerer Rahmen → padding 73
- Internet Explorer 107
- Internet Explorer 3 105
- Internet Explorer 4 105
- Internet Explorer 5.x 106
- Internet Explorer 6 107
- IrfanView 388

J

- Jakob Nielsen 153
- JavaScript 218, 261, 262
- JavaScript Stylesheets 100
- JAWS 95
- JSP 367

K

- Kapitälchen 424
- Kaskadierung 58
- Kastenform 73
- Kastenform → box model 73
- Kastenmodell 120
- Kerning 440
- Kind-Selektor 49
- Kind-Selektor → child 49
- kombinierte Selektoren 232

Kommentar-Trick 126
Kompatibilität 99
Komplettlayout 139
Kontextabhängige Selektoren 47
Konvertierung 89
kursive Schriften 421

L

Längen- und Größenangaben 75
 em 76
 ex 76
 Millimeter 75
 Pica 75
 Pixel 76
 Punkt 75
 Zentimeter 75
 Zoll 75
Lautstärke → volume 530
Layout-Sniffer 379
left 73
legend 301
Leserbereich 543
letter-spacing 440
line-height 443
Linien 297
link relations 279
Linker Abstand → left 73
Listamatic 238
Listen 224, 289
Listenelement 74, 506
List-O-matic 238
List-O-Rama 239
list-style 504
list-style-image 503
list-style-position 504
list-style-type 501
Lycos 307
Lynx 109

M

Macromedia 366, 371
margin 73, 463
 margin-bottom 463
 margin-left 463
 margin-right 463
 margin-top 463
margin-bottom → margin 463
margin-left → margin 463

margin-right → margin 463
margin-top → margin 463
marks 523
Mauszeiger → cursor 515
max-height → height 475
max-width → width 474
media type → Medientyp 256
media-Attribut 71, 125
Medienspezifische Stylesheets 70
Medientyp 256
min-height → height 475
min-width → width 474
Mobile Profile 95
Mobiltelefone 95
Module 83
monospace → Monospace-Schriften 421
Monospace-Schriften 421
Morphen 363
Morphon 363
Mozilla 102
Mozilla-Projekt → Mozilla 102

N

Namensraum 86
namespace → Namensraum 86
Navigation 289, 299
Netscape 4 99, 280
Netscape 6 102
Netscape 7 103
Netscape Navigator 102, 103
Netscape Navigator 6 → Netscape Navigator 102
Netscape Navigator 7 → Netscape Navigator 103
nicht-ersetzte Elemente → non-replaced elements 74
non-replaced elements 74
Nummerierung 501

O

Oberer Abstand → top 73
opacity 497
Opera 107, 108, 112
Opera 4 107
Opera 5/6 107
Opera 7 108
OperaCatcher 126

OpTool 387
Organizer 95
orphans 527
outline 517, 518, 520
outline-color → outline 517
outline-style → outline 517
overflow 477
overflow-x → overflow 477
overflow-y → overflow 477

P

padding 73, 465
 padding-bottom 465
 padding-left 465
 padding-right 465
 padding-top 465
padding-bottom → padding 465
padding-left → padding 465
padding-right → padding 465
padding-top → padding 465
page 527
page-break-after 525
page-break-before 525
page-break-inside 526
Palm 95
Palm-OS → Palm 95
pause 532
PHP 264, 269, 367
pitch 535
pitch-range → pitch 535
Pixel 159, 162, 163
pixelgenaue Layouts 21
play-during 533
position 489, 491
 absolute 170
 fixed 172
 relative 167
 static 165
Positionierung 164, 483, 491, 499
Positionierung → position 489, 491
Positionierungsvarianten 165
Preferred Stylesheets 280
Print 158
projection 280
Prozentangaben 162
prozentuale Angaben 74
Prozentwerte 76
Pseudo-Elemente 54, 395

Pseudo-Formate 395
Pseudo-Klassen 54
Punkte 158

Q

Quellcode 360
Querformat 522
Quirks-Modus 114, 213, 283
quotes 418

R

Rahmen → border 473
Rahmeneigenschaften → border 473
Rechter Abstand → right 493
Referenzkarte 545
Reiter 290
Relative Angaben 158
Relative Einheiten 76
replaced elements 74
richness 536
right 73
Rollover 213
Rollover-Menüs 213
RSS 301

S

Safari 109, 293
sans-serif → serifenlose Schriften 420
Schichten → z-index 498
Schlüsselwörter 74, 78, 156, 162
Schnittmarkierungen 523
Schriftart 420
Schriftfamilie
 generische Schriftfamilie 420
Schriftgestaltung 137
Schriftgröße 156, 163
Schriftgrößen 154, 162, 254, 257
Schriftschnitt 423
Schriftstil 423
Screenreader 94, 289
Seitenabmessungen → size 522
Seitenlayout 521
Selektoren 28, 43, 85
Selektoren-Tricks 126
Selektoren-Wizard 360
Semantik 278
serif → Serifenschriften 420
serifenlose Schriften 420

- Serifenschriften 420
- Sichtbarkeit → visibility 496, 497
- size 522
- Skalierung 163
- Skins 257
- Skip Links 289
- Sliding Faux Columns 193
- SourceForge 382
- speak 529
- speak-header 514
- speak-numeral 539
- speak-punctuation 539
- speech-rate 530
- Spezifität 60
- Sprachausgabe 529
- Sprachgeschwindigkeit → speech-rate 530
- Sprachpausen → pause 532
- Standard Mode 114
- Stilrichtungen 257
- Strategien 135
- stress 536
- Style Master 364
- Style Studio 361
- Stylesheet-Wechsler 257
- Style-Switcher 265
- stylesheet 268
- Suchmaschinen 147
- Suchmaschinenoptimierung 147
- Suchmaschinenoptimierung → Suchmaschinen
- Suckerfish Dropdown 235

T

- Tabellenbeschriftung → caption-side 510
- Tabellenspalten 507
- Tabellenzeilen 507
- table-layout 509
- Tag-Inspektor 368
- Tanteks Hack 157
- Tanteks Hack → Box Model Hack 120, 121
- text-align 445
- text-decoration 434
- Textfarbe 454
- Textfarbe → color 454
- text-indent 444

- text-shadow 293, 438
- text-transform 439
- Thumbnails 240
- Tidy 89, 382
- TidyGUI 382
- top 73, 492
- TopStyle 357, 371

U

- Übersetzung 294
- UltraDev 367
- Umrandung → outline 517, 518, 520
- Unicode 432
- unicode-bidi 453
- Unterer Abstand → bottom 73
- Unterstreichungen 255
- URI 432
- URL 74, 78
- User-Agent 269
- User-Stylesheets 377

V

- Validator → W3C-Validator 136
- Valide Dokumente 135
- Validität 135
- Vektorgrafiken 83
- Vereinfachter Box Model Hack 123
- Vererbung 57
- Verlinkte Stylesheets 67
- vertical-align 446
- Vertikale Menüs 225
- visibility 496, 497
- voice-family 534
- volume 530
- Vordergrundfarbe → color 454

W

- W3C 85
- W3C-Validator 136
- Währungsbezeichnungen 409
- WAI 154
- Web Accessibility Initiative 150, 274
- WebDAV 373, 553
- Webhits 111
- Webspeech 95
- Werbebanner 381
- Wert 28
- Werte für Stylesheets 74

- white-space 442
- widows 528
- width 73, 474
 - max-width 474
 - min-width 474
- WindowsCE 95
- Wireframe 279
- Wizards 361
- word-spacing 441
- WWW 21

X

- x-Höhe → ex 76
- XHTML 85, 86, 367
- XML 28, 90
- XML-Buttons 301
- XML-Prolog 87
- XSL 90

Z

- Zähler 413
- Zeilenhöhe 443
- Zentrieren 193
- Zentrierte Inhalte 193
- Zentrierter Inhalt → Zentrieren
- z-index 498
- Zugängliche Websites 150
- Zugänglichkeit 150
- Zweispaltiges Layout 176