Die Form folgt dem Inhalt

Die Trennung von Form und Inhalt bildet die Grundlage barrierefreier Websites. HTML dient da nur noch der inhaltlichen Strukturierung. Das Design übernehmen Stylesheets.

Eine semantisch korrekte Auszeichnung ist die Grundlage barrierefreien Webdesigns. Doch die Verwendung von HTML-Elementen entsprechend ihrem eigentlichen Verwendungszweck liefert nicht nur Behinderten den Schlüssel zu Ihrem Web-Auftritt. Sie hilft auch Text-Browsern oder Suchmaschinen bei der maschinellen Auswertung Ihrer Online-Inhalte.

In der Praxis galt eine semantisch korrekte Auszeichnung allerdings lange als größtes Hemmnis für „modernes“ Webdesign. Da sich viele HTML-Elemente auch per CSS nur unzureichend formatieren ließen, setzten Webdesigner auf so genannte Tabellen-Hacks. Dabei bestimmten unsichtbare Tabellen nicht selten den gesamten Seitenaufbau. Diese damals wirkungsvolle Technik widersprach allerdings nicht nur dem Grundsatz der Trennung von Inhalt und Layout, sondern führte im Laufe der Zeit selbst auf bedeutenden Online-Präsenzen zu einem kaum noch zu entwirrenden Wildwuchs im HTML-Code.

Selbst heute erfolgt beispielsweise das Anordnen von Layout-Blöcken häufig noch über Tabellen. Deren semantisch korrekter Verwendungszweck beschränkt sich allerdings auf die Darstellung tabellarischer Daten. Ihr Einsatz als Layout-Tabellen ist nichts weiter als eine sinnlose Zweckentfremdung, welche beim Redesign einen erhöhten Arbeitsaufwand verursacht und sehbehinderten Menschen, die im Web auf den Einsatz eines Screenreaders angewiesen sind, das Leben nur unnötig erschwert.

Linearisierung. Screenreader sind spezielle Ausgabegeräte für Blinde und stark Sehbehinderte, die den Text auf einem Computer-Bildschirm zeilenweise vorlesen. Tabellen gibt ein solcher Screenreader im Allgemeinen linearisiert aus, das heißt die Inhalte werden von links nach rechts und von oben nach unten gelesen. Eine derartige zeilenweise Ausgabe muss allerdings nicht unbedingt der optischen Darstellung entsprechen, und so kommt es gerade bei ineinander verschachtelten Tabellen mehrspaltiger Layouts schnell zu völlig unsinnigen Screenreader-Ausgaben.

Um der linearisierten Zugriffsweise von Screenreadern und ähnlichen Hilfsmitteln Rechnung zu tragen, fordert die Barrierefreie Informationstechnik-Verordnung (BITV): „Tabellen sind nicht für die Text- und Bildgestaltung zu verwenden, soweit sie nicht auch in linearisierter Form dargestellt werden können.“ Dass sich reine Layout-Tabellen ohne weiteres durch eine linearisierte

Überflüssige Tabellen: Ähnliches Layout, doch T-Online verwendet rund 250 Tabellenzellen, während GMX ganz auf Tabellen verzichtet

kurzübersicht

BARRIEREFREIHEIT

Die Form folgt dem Inhalt – das gilt insbesondere für barrierefreies Webdesign. Erst eine semantisch korrekte Auszeichnung im HTML-Quelltext macht Ihre Online-Inhalte allen Anwendern zugänglich.

Darstellungsform ersetzen lassen, zeigt Ihnen die IW im Folgenden anhand eines einfachen, zweispaltigen Layouts mit einem Logo und einer Hauptnavigation im Seitenkopf, einer Sub-Navigation am linken Bildschirmrand und der Content-Darstellung im verbleibenden Seitenbereich rechts unten.

Grund-Layout. Zur Gliederung der verschiedenen Layout-Blöcke verwenden wir für unser linearisiertes Design mehrere *div*-Container: Der Seitenkopf (*header*) umfasst das Site-Logo (*logoarea*) sowie die Hauptnavigation (*main-nav*), während sich der Hauptbereich der Seite (*main*) aus der Sub-Navigation (*sub-nav*) sowie dem eigentlichen Seiteninhalt (*content*) zusammensetzt.

```
<div id="header">
 <div id="logoarea">Logo</div>
 <div id="main-nav">Hauptmenü</div>
</div>
<div id="main">
 <div id="sub-nav">Sub-Menü</div>
 <div id="content">Inhalt</div>
</div>
```

Die gesamte Formatierung überlassen wir einer extern eingebundenen CSS-Datei. Hier der entscheidende Auszug für die Formatierung des Hauptbereichs:

```
#main{
 width:100%;
 background: #eee;
 border-top:0.2em solid #046;
 border-bottom:0.2em solid #046;
}
#sub-nav{
 width:13em;
 float:left;
}
#content{
 margin: 0 0 0 13em;
 padding: 0 0 0 1em;
 background:#fff;
}
```

Grundlegend ist hierbei, dass die Sub-Navigation mittels *float:left* als Floating-Bereich definiert wird. Ohne weitere Angaben würde der folgende Content-Bereich die Sub-Navigation also umfließen. Da wir aber dem Content-Bereich einen linken Rand (*margin: 0 0 0 13em*) zuweisen, der ebenso groß definiert ist wie die Breite der Sub-Navigation (*width:13em*), unterbinden wir dies.

Als Hintergrundfarbe für den Hauptbereich verwenden wir die gewünschte Hintergrundfarbe der Sub-Navigation. Dadurch wird diese „linke Spalte" der Seite auch dann komplett farbig hinterlegt, wenn ihr Inhalt kürzer ist als der des Content-Bereichs. Das Resultat entspricht dann dem gewohnten 2-Spalten-Layout vieler Websites.

Logo-Zoom. Die Gestaltung des Seitenkopfs erfolgt völlig analog. Auch hier verwenden wir bei allen Größenangaben relative *em*-Werte, die sich auf die jeweils aktuelle Schriftgröße beziehen. Dadurch bleiben bei Verwendung größerer Schriftgrade alle Dimensionen erhalten und selbst die Linien über- und unterhalb des Hauptbereichs (*border-top* und *border-bottom*) der Seite werden auf diese Weise zoombar.

Selbstverständlich lassen sich auch die Größenangaben von Bildern in *em*-Werten angeben. Aus folgender Bild-Definition ▶

webcode 0412056

WEB-ACCESSIBILITY

Geben Sie auf ***www.internetworld.de*** den **Webcode 0412056** ein.
Sie gelangen zu folgenden Angeboten:

- Beispiel-Listings zu diesem Workshop
- CSS Templates
- Floatutorial: HTML-Elemente umfließen
- Listutorial: HTML-Listen als Menüs
- Tastaturbedienung und Tabindex
- Empfehlungen für Shortcuts auf Web-Seiten
- W3C-Infos zum link-Element
- Browser mit link-Unterstützung
- Checklisten gemäß BITV
- Screenreader-Simulator
- Lynx Viewer (Text-Browser)

Zoombar: Bei der Wahl eines größeren Schriftgrads bleiben die relativen Dimensionen des Logos erhalten

```
<div id="logoarea">
 <img id="logo" src="logo.gif"
 alt="Logo der Zeitschrift INTERNET WORLD">
</div>
```

machen Sie in der CSS-Datei durch relative Größenangaben von *width* und *height* ein zoombares Site-Logo:

```
#logo{
 width:13em;
 height:4em;
}
```

Um dabei Verzerrungen zu vermeiden, sollten Sie allerdings die relativen Dimensionen entsprechend den Seitenverhältnissen Ihrer Logo-Grafik wählen. Oder andersherum: Definiert Ihr Grund-Layout eine relative Größe, so muss die tatsächliche Größe – bei gleichem Seitenverhältnis – ein Vielfaches davon betragen. Zudem ist es ratsam, die Grafik etwas größer anzulegen, damit sie auch in größeren Darstellungen sauber dargestellt wird. In unserem Beispiel verwenden wir eine Grafik mit 650 x 200 Bildpunkten, was dem 50fachen Wert der relativen *em*-Angaben entspricht.

Hover-Menüs. Tabellen wurden in der Vergangenheit nicht nur zur Platzierung ganzer Layout-Blöcke eingesetzt, sie kamen auch sehr oft beim Aufbau aufwendiger Navigationsmenüs zum Einsatz. Noch heute setzen große Website wie *focus.de* oder *t-online.de* Layout-Tabellen in ihren Navigationselementen ein. Einer semantisch korrekten Auszeichnung entspricht das allerdings nicht.

Die aktuellen HTML-Standards stellen bislang keine spezifischen Elemente für Navigationsmenüs bereit. In der Praxis haben sich allerdings unnummerierte Listen zur Gliederung von Menüstrukturen bestens bewährt. Sie geben die Informationsarchitektur einer Website am besten wieder, da sie die einzelnen Menüeinträge klar voneinander trennen und sich Untermenüs über verschachtelten Listen leicht abbilden lassen. CSS-Formatierungen sorgen zudem für ansprechendes Design und Hover-Effekte ohne JavaScript-Code.

Im folgenden HTML-Fragment verwenden wir eine einfache, unnummerierte Link-Liste für das Untermenü unserer Beispielseite:

```
<ul id="sub-navigation">
 <li><a href="#">Aktuelles Heft</a></li>
 <li><a href="#">Nächstes Heft</a></li>
 <li><a href="#">Heft-Archiv</a></li>
 <li><a href="#">Abonnement</a></li>
</ul>
```

Aus dieser Liste soll nun ein vertikales Menü entstehen, das auf die üblichen Aufzählungszeichen einer Liste verzichtet, dafür aber mit Hover-Effekten aufwarten kann:

```
#sub-navigation{
 list-style:none;
 margin:0;
}
```

```
#sub-navigation a{
 display:block;
 color: #046;
 text-decoration: none;
}
#sub-navigation a:hover,
#sub-navigation a:focus,
#sub-navigation a:active{
 color: #fd0;
 background:#046;
}
```

Wenden Sie *list-style:none* auf das *ul*-Element an, verschwinden die Aufzählungszeichen der Liste und mit *margin:0* auch die übliche Einrückung der einzelnen Listenelemente. Für den Hover-Effekt reicht es, für die Pseudoelemente *a:hover, a:focus* und *a:active* eine andere Vorder- und Hintergrundfarbe zu wählen. Die Anweisung *display:block* sorgt schließlich dafür, dass sich der Hover-Effekt auf der gesamten Breite auswirkt und nicht nur auf die Länge der Link-Bezeichnung beschränkt bleibt. Wollen Sie ein derartiges Menü hingegen horizontal anordnen, dann setzen Sie zusätzlich ein Floating für das *li*-Element:

```
#sub-navigation li{
 float:left;
}
```

Für die nötigen Abstände zwischen den einzelnen Menüeinträgen sorgen schließlich entsprechende *margin*- und *padding*-Angaben für die Listeneinträge und Links.

Tabbing & Accesskeys. In den Links unseres Beispiel-Menüs fehlen bislang nicht nur die Verweisziele, sondern auch noch einige Attribute, die weitere Barrieren aus dem Weg räumen. Da wäre zunächst das *title*-Attribut, dessen Angabe bei den meisten Browsern als Tooltip eingeblendet wird. Hier lassen sich nähere Details zur Zielseite un-

Ohne Script: Hover-Effekte lassen sich dank CSS auch ohne aufwendige JavaScripts erzeugen

terbringen, die zum Beispiel auch weniger erfahrenen Internet-Benutzern vermeintlich unbekannte Link-Bezeichnungen wie etwa „Sitemap“ erläutern.

Für Endgeräte ohne Maus, Trackball oder ein entsprechendes Eingabegerät lässt sich über das Attribut *tabindex* zudem für die Tastaturbedienung eine Tabulator-Reihenfolge der Verweise festlegen. Springt man im Webbrowser mithilfe der Tabulator-Taste nacheinander die Verweise einer HTML-Datei an, so werden diese normalerweise in der Reihenfolge durchlaufen, in der sie im Quelltext der Seite definiert sind. Mittels *tabindex* lässt sich diese Reihenfolge nun verändern. Ein kompletter Link sähe damit folgendermaßen aus:

```
<a href="/"
title="Zur Startseite der INTERNET
WORLD Website" tabindex="1"
>Start</a>
```

Kontraproduktiv kann die Verwendung des *tabindex* allerdings bei gleichzeitigem Einsatz so genannter Skip-Links werden, die es dem Anwender beispielsweise erlauben, direkt zum Inhalt der Seite zu springen (siehe Webcodes Seite 57: Tastaturbedienung und Tabindex). Das Problem: Einige Webbrowser setzen die Tab-Reihenfolge nicht am internen Sprungziel fort, sondern halten sich stattdessen sklavisch an den festgelegten Tabindex. Dann kann es passieren, dass der Benutzer nach einem Sprung zum Inhalt beim nächsten Tab wieder im Hauptmenü der Seite landet und nicht beim ersten Link innerhalb des Inhalts.

Noch problematischer ist die Verwendung von Tastaturkürzeln, die sich über das Attribut *accesskey* festlegen lassen. Hier variiert schon der Aufrufmechanismus von Browser zu Browser. So wird *accesskey = “x“* unter Windows beim Internet Explorer und Netscape-/Mozilla-Browser über die Tastenkombination [Alt] + [x] aktiviert, allerdings erwartet der IE zusätzlich ein Return. Opera 7+ hingegen verwendet die Tastenkombination [Shift] + [Esc] + [x], und unter Mac Os verlangen Mozilla & Co nach der Ctrl-Taste. Darüber hinaus können Accesskeys auch die Funktionalität des Browsers einschränken. So versperrt *accesskey = “d“* beispielsweise im Internet Explorer den Tastaturzugriff auf das Dateimenü.

Andere Browser und andere Sprachvarianten kennen wiederum andere Tastaturkürzel für ihre Grundfunktionen. Als relativ unproblematisch gelten deshalb lediglich Accesskeys mit den Ziffern 0 bis 9, weshalb wir in unserem Beispiel gleich ganz auf Tastaturkürzel verzichten.

Links im Kopfbereich. Eingangs erwähnten wir, dass die aktuellen HTML-Standards bislang keine spezifischen Elemente für Navigationsmenüs bereitstellen. Nun, das stimmt nicht ganz. Bereits der HTML-2.0-Standard vom November 1995 enthielt ▶

das *link*-Element, über das sich Verweise zu Standardseiten definieren lassen, die von den meisten Websites benutzt werden: die Homepage, eine Online-Hilfe, die nächste beziehungsweise vorherige Seite oder aber ein Dokument mit Kontaktinformationen.

Mozilla und der Opera Browser unterstützen diese standardisierten „Verkehrshinweise" von Haus aus über eine eigene Symbolleiste, während bei Microsofts Internet Explorer und dem Firebird Browser die Installation eines speziellen Plug-ins erforderlich ist. Definiert werden die Standard-Links, ähnlich wie externe Stylesheets, im *head*-Bereich der HTML-Seite:

```
<link rel="home"
href="http://www.internetworld.de"
title="Startseite">
<link rel="copyright"
href="http://www.internetworld.de/
sixcms/detail.php?template=iw_
impressum" title="Impressum">
<link rel="author"
href="mailto:mail@internetworld.de"
 title="Mail an die Redaktion">
```

Der Vorteil derartiger Standard-Links liegt auf der Hand: Hat sich der Benutzer erst einmal an die entsprechenden Symbol-Leisten seines Webbrowsers gewöhnt, fällt ihm auch auf bislang unbekannten Internet-Angeboten die Orientierung deutlich leichter. Zudem werten moderne Webbrowser derartige Angaben auch für andere Zwecke aus. So nutzt der Firebird Browser beispielsweise eine entsprechende *link*-Angabe, um den Benutzer auf die Verfügbarkeit eines RSS-Newsfeeds hinzuweisen und ihm diesen als Live-Bookmark anzubieten.

Umformatiert: Alternative Ansichten, zum Ausdrucken oder für Graustufen-Monitore, lassen sich dank CSS schnell erstellen

Ansichtssache. Eine semantisch korrekte Auszeichnung der HTML-Datei und die konsequente Auslagerung von Layout-Anweisungen in CSS-Dateien erfreut auch die Suchmaschinen. Deren Robots müssen sich nicht länger durch einen riesigen Ballast von Layout-Anweisungen quälen und können relevante Textstellen – etwa Überschriften – dank korrekter Auszeichnung leichter erkennen.

Doch auch bei etwaigen Layout-Änderungen oder Anpassungen für weitere Ausgabegeräte bewährt sich die strikte Trennung von Form und Inhalt. So können Sie zum Beispiel über alternative Stylesheets schnell weitere Layout-Varianten einbinden, und selbst ein komplettes Redesign lässt sich nun ausschließlich über die ausgelagerten CSS-Dateien realisieren.

In unserem Muster-Layout (siehe Webcodes) verwenden wir beispielsweise alternative Stylesheets für eine Seitenansicht in Graustufen sowie für eine Druckansicht. Letztere macht sich übrigens nicht nur auf dem Papier gut: Wählt der Benutzer einen größeren Schriftgrad so lässt sich eine Druckansicht im Web dank relativer Größenangaben selbst bei starker Sehbehinderung noch immer gut konsumieren. ■ Stefan Kuhn

info

DIE 14 ECKPUNKTE DER BITV-CHECKLISTE ZUR BARRIEREFREIEN WEB-GESTALTUNG

1. Für jeden Audio- oder visuellen Inhalt sind geeignete äquivalente Inhalte bereitzustellen, die den gleichen Zweck oder die gleiche Funktion wie der originäre Inhalt erfüllen.
2. Texte und Grafiken müssen auch dann verständlich sein, wenn sie ohne Farbe betrachtet werden.
3. Markup-Sprachen (insbesondere HTML) und Stylesheets sind entsprechend ihrer Spezifikationen und formalen Definitionen zu verwenden.
4. Sprachliche Besonderheiten wie Wechsel der Sprache oder Abkürzungen sind erkennbar zu machen.
5. Tabellen sind mittels der vorgesehenen Elemente der verwendeten Markup-Sprache zu beschreiben und in der Regel nur zur Darstellung tabellarischer Daten zu verwenden.
6. Internet-Angebote müssen auch dann nutzbar sein, wenn der verwendete Benutzer-Agent neuere Technologien nicht unterstützt oder diese deaktiviert sind.
7. Zeitgesteuerte Änderungen des Inhalts müssen durch den Nutzer der Website kontrollierbar sein.
8. Die direkte Zugänglichkeit der in Internet-Angeboten eingebetteten Benutzerschnittstellen ist sicherzustellen.
9. Internet-Angebote sind so zu gestalten, dass Funktionen unabhängig vom Eingabegerät oder Ausgabegerät nutzbar sind.
10. Die Verwendbarkeit von nicht mehr dem jeweils aktuellen Stand der Technik entsprechenden, assistiven Technologien und Browsern ist sicherzustellen, soweit der hiermit verbundene Aufwand nicht unverhältnismäßig ist.
11. Die zur Erstellung des Web-Angebots verwendeten Techniken sollen öffentlich zugänglich und vollständig dokumentiert sein, wie etwa die vom W3C entwickelten Technologien. Die Verwendung von Funktionen, die durch die Herausgabe neuer Versionen überholt sind, ist zu vermeiden.
12. Dem Nutzer des Online-Angebots sind Informationen zum Kontext und zur Orientierung bereitzustellen.
13. Navigationsmechanismen sind übersichtlich und schlüssig zu gestalten.
14. Das allgemeine Verständnis der angebotenen Inhalte ist durch angemessene Maßnahmen zu fördern.