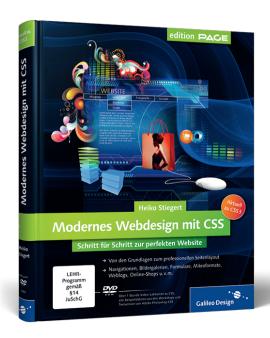
# Modernes Webdesign mit CSS

Schritt für Schritt zur perfekten Website





# Inhalt

Vorv	vort		5
TEI	L I G	rundlagen	
1	Einle	eitung	
1.1		st benutzerfreundliches Webdesign?	15
	1.1.1	Accessibility	15
	1.1.2	Usability	16
1.2	Was i	st erfolgreiches Webdesign?	17
	1.2.1	Suchmaschinenoptimierung – SEO (Onpage)	17
	1.2.2	Konversion	18
2	CSS-	-Basiswissen	
2.1	Was i	st CSS?	21
	2.1.1	Funktionsweise von CSS	22
2.2	CSS i	n HTML einbinden	23
		Einbindung per »style«-Attribut	23
	2.2.2	Einbindung per »style«-Element	23
	2.2.3	Einbindung einer externen CSS-Datei per	
		HTML-Anweisung	24
	2.2.4	Einbindung einer Datei per CSS-Anweisung	25
	2.2.5	Einbindung einer CSS-Datei für den Internet Explorer:	
		Conditional Comments	25
2.3	CSS-F	Regeln	27
	2.3.1	Aufbau von CSS-Regeln	27
2.4	Name	enskonventionen	31
2.5	Maße	inheiten	33
	2.5.1	Absolute Größenangaben	34
	2.5.2	Relative Größenangaben	35
	2.5.3	Schlüsselwörter	37
2.6	Inline	- und Blockelemente	38
	2.6.1	Blockelemente	38
	2.6.2	Inlineelemente	39
	2.6.3	Inline-Block-Elemente	41
2.7	Das B	ox-Modell	41



2.8	Positi	onieren und Stapeln	43	
	2.8.1	Relative Positionierung	43	
	2.8.2	Absolute Positionierung	44	
2.9	Floati	ng und Clearing	47	
	2.9.1	»float« zuweisen	48	
	2.9.2	»float« mit »clear« aufheben	50	
2.10	Progr	essive Enhancement und Graceful Degradation	52	
	2.10.1	Was bedeutet »Progressive Enhancement«?	53	
	2.10.2	Was bedeutet »Graceful Degradation«?	54	
3	Layo	uttypen		
3.1	Fixes	Design (fix Layout) mit fester Breite	55	
3.2	Fließendes Design (fluid Layout)			
3.3				
3.4		oformen	59 62	
			-	
4	Seite	enoptimierung und Debugging		
4.1	Web-	Performance-Optimierung (WPO)	63	
	4.1.1	HTML-Performance verbessern	63	
	4.1.2	CSS-Performance verbessern	64	
	4.1.3	Performance der Bilder verbessern	66	
	4.1.4	Load Sharing	70	
4.2	Debu	gging und Analyse	71	
	4.2.1	CSS- und HTML-Validierung	71	
	4.2.2	Debugging mit Browser-Extensions	72	
	4.2.3	Mehrere Browserversionen	78	
	4.2.4	Virtuelle Maschinen – Parallels, VMware & Co	79	
	4.2.5	Screenshot-Services	80	



## **TEIL II Seitenelemente**

#### Die Site strukturieren 5

<b>Übersichtlich gegliedert: Navigationen gestalten</b> Eine benutzerfreundliche Tab-Navigation mit und ohne Grafiken	84
Ganz ohne Grafiken: Slogans, Logos, Aufmacher  Mit individuell gestalteten Überschriften die  Aufmerksamkeit des Besuchers lenken	96
Von Kopf bis Fuß: der Webseitenfooter  Webseitenfooter am Boden des Browserfensters fixieren	104





#### Bilder einsetzen 6

Professionell präsentiert: Bildergalerien im Web  Eine klassische Lightbox ohne JavaScript	114
Sprechblasen und Teaserboxen  Text und Bild clever kombinieren	125
Zitate individuell gestalten  Mit ansprechend gestalteten Kundenbewertungen  Vertrauen aufbauen und Authentizität schaffen	136
Initialen und mehrspaltige Layouts Einen Text im Magazinlayout gestalten	145
7 Daten visualisieren und eingeben	
Daten übersichtlich präsentieren: Tabellen Einzelne Tabellenspalten optisch hervorheben	158
Onlineformulare benutzerfreundlich gestalten	182

# **TEIL III Trends, Tipps & Tricks**



#### 8 Weblogs

Parallax-Effekt im Header  Aufmerksamkeit durch interaktive Bewegung	210
Ein Blogdesign im Retro-Stil »Retro« als Gestaltungsmittel	226
Kommentardesign in Blogs  Mit ansprechend gestalteten Kommentaren  zum besseren Blogdesign	242

#### Firmen- und Freelancer-Sites 9

G	ileich hohe »div«-Elemente für alle Browser	262
	Eine Mindesthöhe für Infoboxen mit	
	unterschiedlichen Inhalten	
ŀ	hre Visitenkarte im Netz: ein Portfolio gestalten	280
	Einblend-Effekte und Slide-Animationen für	
	eine gelungene Präsentation	

# 10 Onlineshops

Breadcrumb-Navigationen		
Den Weg des Besuchers auf der Website kennzeichnen		
Umfangreiche Onlineformulare	315	
Verbesserte Benutzerführung für umfangreiche		
Bestell- oder Registrierungsformulare		



## TEIL IV Web 3.0

# 11 CSS3 für eine kreative Gestaltung

Coffee-Cards – eine etwas andere Liste  Kreative Gestaltung mit CSS3: Farbverlauf,  Neigung, Schatten, runde Ecken	346
Akkordeon-Effekt  Elemente ein- und ausblenden mit CSS3-Pseudoklassen – ohne JavaScript	362
Responsive Webdesign mit Media Queries  Eine Website für mobile Endgeräte optimieren	375
Webdesign im Miniaturformat: mobile Websites	386



## 12 Das semantische Web

Kontaktdaten effektiv einsetzen mit hCard	400
Fermine formatieren mit hCalendar  Auslesen und Speichern von Veranstaltungsdaten direkt aus dem Browser	412
nReview: Bewertungen maschinenlesbar umsetzen	424

Die DVD zum Buch	439
Index	441

#### Layouttypen 3

Mit der Gestaltung Ihrer Webseiten bestimmen Sie, wie die Text-, Bildund Videoinformationen vom Nutzer aufgenommen werden. Eine wesentliche Rolle spielt dabei der Viewport, also die Größe des Browserfensters. Im Idealfall nutzt eine Website den zur Verfügung stehenden Platz so aus. dass alle Inhalte gut zugänglich sind.

#### Fixes Design (fix Layout) mit fester Breite 3.1

Für die Anordnung der verschiedenen Bereiche und Inhalte einer Website gibt es verschiedene Ansätze. Einer davon ist das sogenannte fixe Layout. Damit bezeichnet man eine Variante der Layoutgestaltung, die auf absoluten Breiten- und Abstandsangaben basiert.

```
#wrapper { width:950px; }
```

Neben den allumfassenden Containern werden auch die einzelnen Elemente einer Webseite, wie in Abbildung 3.1 eine Sidebar, die Formularoder Gestaltungselemente, mit absoluten Breiten- und Abstandsangaben in Pixeln innerhalb ihrer so erzeugten Grenzen ausgerichtet.

Das bedeutet, ein zentriert ausgerichtetes Webseitenlayout von beispielsweise 950 px in der Breite wird bei einem Viewport mit einer Breite von knapp 1.000 px ebenso breit dargestellt wie bei einem Viewport mit einer Breite von 1.600 px (siehe Abbildung 3.2).

So schön und unter Umständen einfach auch die Umsetzung eines pixelgenauen Entwurfes ist - sobald der Anwender die Schriftgröße über die entsprechende Browserfunktion ändert, verändert sich auch das Layout.

Beim Page-Zoom (Seitenzoom), den mittlerweile alle aktuellen relevanten Browserversionen ohne Probleme umsetzen können (siehe rechten Teil der Abbildung 3.3), stellt dies dagegen kein wirkliches Problem dar, weil hier das gesamte Layout verändert wird.

#### Das fixe Layout

Diese Variante der Layoutgestaltung basiert auf absoluten Pixelangaben für die Elemente einer Webseite und deren gesamter Ausbreitung. Somit ist ein solch umgesetztes Seitenlayout unabhängig vom ieweiligen Viewport des Besuchers der Webseite. Das bedeutet, die Breite einer Webseite von beispielsweise 760 Pixeln wird auf einem Viewport (Größe des Browserfensters) von 1.000 Pixeln ebenso wie bei einem Viewport von 1.600 Pixeln dargestellt.

#### Vorteile eines fixen Layouts

- Webseiten mit einem fixen Lavout sind leichter zu designen und umzusetzen als fließende oder elastische Designs.
- Die einzelnen Bereiche und Elemente bleiben in ihrer Anordnung und in ihren Maßen unabhängig vom Viewport gleich.





#### ▲ Abbildung 3.1

Fixes Layout bei einem Viewport von 1.000 px

#### ▲ Abbildung 3.2

Fixes Layout bei einem Viewport von 1.600 px

#### Page-Zoom

Folgende Browser-Versionen können den Page-Zoom korrekt umsetzen: Firefox ab Version 3, Google Chrome ab Version 1, Safari ab Version 4, Opera ab Version 8 und der Internet Explorer ab Version 7.

Das heißt, in dieser hier verwendeten Beispielabbildung des aus Kapitel 10 stammenden Formular-Workshops skalieren sowohl die Hintergrundgrafik als auch die absolut ausgerichteten Inhalte und die Illustration. Verändert der Anwender allerdings lediglich die Schriftgröße (siehe linken Teil von Abbildung 3.3), ändern sich nur die textlichen Inhalte, womit das Layout in diesen Bereichen nicht mehr das gewünschte Verhältnis zwischen Text und den dahinter- oder umliegenden Elementen aufweist.





#### ▲ Abbildung 3.3

Vergleich zwischen Text- (links) und Seitenzoom (rechts)

Drei Beispielwebseiten, die auf dieser Art der fixen und absoluten Layoutgestaltung basieren, sind in Abbildung 3.4 zusammengefasst.



#### ▲ Abbildung 3.4

Webseiten mit absolutem Layout (von links nach rechts): Veerle's blog (veerle.duoh.com), MailChimp (mailchimp.com) und hey Indy (heyindy.com)

#### Fließendes Design (fluid Layout) 3.2

Designs von Webseiten müssen allerdings nicht immer auf absoluten Maßangaben beruhen, sondern können auch mittels relativer Angaben eine Flexibilität aufweisen, die es ihnen ermöglicht, sich dem Viewport des Betrachters anzupassen. Ausschlaggebend für ein solches Verhalten sind in erster Linie die Maßangaben, die sich der Breite des Gesamtlayouts und der darin befindlichen Elemente widmen. Diese kleinen, aber wirksamen Änderungen innerhalb der CSS-Angaben führen dazu, dass sich die mit Prozentangaben ausgestatteten Bereiche eines Webseitenlayouts bei der Größenänderung des Browserfensters entsprechend anpassen.

In den Abbildungen 3.5 und 3.6 sehen Sie ein Beispiel aus Kapitel 5, das mit einem solchen fließenden Layout umgesetzt wurde. Der allumfassende wrapper nutzt 75% der vorhandenen Viewportbreite.

#wrapper { width:75%: }

Die für ein fließendes Layout angegebenen prozentualen Breiten- und Abstandsangaben erlauben somit ein Höchstmaß an Flexibilität. Wie in dem hier vorgestellten Workshop-Beispiel kann der Viewport allerdings so stark verkleinert werden, bis das Layout einen horizontalen Scrollbalken erhält, weil beispielsweise die darin enthaltene Überschrift nicht mehr den notwendigen Platz besitzt, sich entsprechend ihrer Eigenschaften ausbreiten zu können. Bei einem fixen Layout würde der Scrollbalken eines 950 px

#### Nachteile eines fixen Lavouts

- Wird ausschließlich die Schriftgröße verändert, kann dies dazu führen, dass Texte aus dem Lavout oder aus in der Breite eingegrenzten Elementen herauslaufen und das Layout »sprengen«.
- Ein pixelgenaues Layout ist für eine bestimmte Viewportgröße optimiert und kann sich den zahlreichen unterschiedlichen Endgeräten mit ihren unterschiedlichen Viewportgrößen nicht anpassen.

#### Das fließende Layout

Fließende Layouts sind seit Beginn des Internes Bestandteil der Gestaltung von Webseiten, denn von Haus aus sind Blockelemente wie div-Container. Überschriften oder Textabsätze ohne Breitenangaben mit einer flexiblen Breite versehen, die sich entsprechend dem zur Verfügung stehenden Platz ausdehnt. Ein solches Layout basiert somit primär auf der Breite des Viewports. Das heißt, eine relative Breitenangabe von 75 % für den gesamten Auftritt wird bei einem Viewport von 1.000 px mit einer Breite von 750 px angezeigt, wesentlich schmaler als bei einem Viewport von 1.600 px, wo die Webseite 1.200 px breit dargestellt würde.

#### Vorteile des fließenden Layouts

- ► Fluid Layouts passen sich dem Viewport des Ausgabemediums
- Textinhalte von Webseiten dieses Layouttyps können sich der Größe des Viewports anpassen.
- Horizontales Scrollen kann unabhängig vom Viewport vermieden werden.
- Eine Anpassung der Breiten des Designs und der darin enthaltenen Elemente ist nicht zwingend notwendig, da das Layout sich auch kleinen Viewports anpasst.

Detail Bearbeiten Amsicht Verlauf Lesszeichen Entwickler

In Heine Congressen Entwickler

See how it works

Dus autem vel eum kürze cklor in hendrent in vulputate veit esse moleste consequet, veit kum deben ein festjalt in sich seitige ein versicht versichen ein festjalt in sich seitige ein versicht versichen der versichten versichen der versichten der versichten versichen der versichten versichen der versichten versichen der versichten versichten

### ▲ Abbildung 3.7

Fluid Layout bei einem Viewport von 400 px

breiten Layouts hingegen bereits bei 949 px erscheinen, obwohl die Elemente immer noch genügend Platz besäßen.



#### ▲ Abbildung 3.5

Fluid Layout bei einem Viewport von 1.000 px



#### ▲ Abbildung 3.6

Fluid Layout bei einem Viewport von 1.600 px

Neben den Breitenangaben für allumfassende Container werden auch die Breiten der darin enthaltenen Elemente in Prozent angegeben. Werden die Breitenangaben weggelassen, breiten sich Blockelemente immer in Abhängigkeit von dem ihnen zur Verfügung stehenden Platz aus. Das ist durch ein width: auto in den Browser-Defaultstylesheets geregelt. In diesem Fall bedeutet das, dass sich beispielsweise die beiden Textspalten der Abbildun-

gen 3.5 und 3.6 ie nach Viewport verkleinern oder verbreitern. Schränken Sie allerdings die maximale Ausdehnung nicht ein, kann ein fließendes Layout durchaus zu einem Usability-Problem führen: Die Texte laufen zu breit und sind nicht mehr lesbar. Eine Möglichkeit, das zu umgehen, ist die Eigenschaft max-width. Lesen Sie dazu bitte auch Abschnitt 3.4. »Mischformen«.

Zudem müssen Sie bei diesem Lavouttyp unbedingt darauf achten, dass Webseiten auch bei einem kleinen Viewport von beispielsweise 400 px wie in Abbildung 3.7 weiterhin handhabbar sind und nicht durch horizontale und vertikale Scrollbalken gänzlich unübersichtlich werden. In Kombination mit CSS3 Media Queries (mehr dazu im Workshop »Responsive Webdesign mit Media Queries« in Kapitel 11) können Sie einem fließenden Layout für unterschiedliche Viewportgrößen den Anforderungen entsprechend individuelle Breitenabgaben zuweisen und unter Umständen auch die dann nicht mehr notwendige Spalteneigenschaften des Fließtextes aufheben.

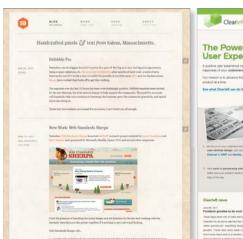
Da auch diese Art von Layout von zahlreichen Webseitenbetreibern als Basis zur Gestaltung des eigenen Lavouts herangezogen wird, zeigt Abbildung 3.8 nun drei Beispielwebseiten, die auf ebendiese Art des fließenden Layouts zurückgreifen.

#### Nachteil des fließenden Layouts

Pixelgenaues Ausrichten von Elementen ist aufgrund der unterschiedlichen Viewports der unterschiedlichen Endgeräte der Anwender nicht möglich.

#### min-width und max-width

Auch bei fließenden Layouts können und sollten Sie Grenzen setzen: Denken Sie daran für minwidth und max-width Werte zu vergeben, denn bei sehr kurzen oder sehr langen Zeilen leidet die Lesbarkeit.







#### **Elastisches Design (elastic Layout)** 3.3

Ein Layout basierend auf den relativen Größenangaben % (für die Schriftgrößen) und em (für die Elemente an sich) wird zu einem sogenannten elastischen Layout. »Elastisch« heißt in diesem Fall, dass es in Bezug auf Höhe, Breite und Abstände skalierbar ist.

Der Vorteil dieses Layouttyps gegenüber einem fixen Layout liegt darin, dass bei der Veränderung der Schriftgröße durch den Anwender nicht nur

#### ▲ Abbildung 3.8

Webseiten mit fluid Layout (von links nach rechts): SimpleBits (simplebits.com), Clearleft (clearleft.com) und dcc design (dccdesign.co.uk)

#### Das elastische Layout

In einem elastischen Layout werden alle Größenangaben für sämtliche Elemente mit der relativen Maßeinheit em ausgezeichnet. Diese Flemente verhalten sich proportional zu den in Prozenten angegebenen Schriftgrößen. Wird die in der Browsereinstellung festgelegte Schriftgröße verändert, skalieren sämtliche Seitenelemente analog zur Veränderung mit. Das heißt, die Größe der Elemente ändert sich, aber das Größenverhältnis – sprich die Proportionen – bleibt gleich. Ein einfaches em-skalierendes Layout erreicht man schon, wenn man dem #wrapper eine em-Breite zuweist und alle inneren Container weiter mit %-Werten behandelt Bekannt wurde das elastische Lavout Ende 2003, als es in einem Layout des »CSS Zen Gardens« (www.csszengarden.com/063) den Webdesignern von Patrick Griffiths vorgestellt und anschließend in einem Beitrag auf »A List Apart« beschrieben wurde: www.alistapart.com/articles/ elastic

#### Vorteile des elastischen Layouts

Der Vorteil der elastischen Layouts ist die erhöhte Zugänglichkeit: Denn wenn der Anwender im Browser die Schriftgröße ändert, verändert sich das gesamte Layout abhängig von dieser Schriftgrößenanpassung. Das heißt, eine Vergrößerung des Textes sorgt für eine Vergrößerung des Layouts und umgekehrt, und das, ohne die Proportionen der Inhalte und Webseiten-Elemente zueinander zu verlieren, wie es bei einem fixen oder fließenden Layout der Fall wäre.

Textinhalte, sondern auch die dazugehörigen Elemente, wie Bilder und multimediale Inhalte, mitskalieren. Verwenden Sie diese relative Maßangabe für die initiale Schriftgrößenangabe, verhalten sich die innerhalb dieses Elements befindlichen Inhalte (#wrapper . . .) relativ zu diesem Elternelement.

Aufgrund der damit verbundenen Flexibilität ist aber der Arbeitsaufwand größer als beim fixen Layout. Ausschlaggebend für die Maße der Elemente ist die initial für das body-Element gesetzte Schriftgröße, denn dies ist der Wert, den der Anwender seinen Anforderungen entsprechend verändert.

body { font-size:100%; }

Größenangaben in em verhalten sich proportional zur Text- oder Schriftgröße, wenn diese ebenfalls relativ (em oder %) angelegt wurde. Wird über die Browsereinstellung der Schriftgrad verändert, erhöhen sich auch die Bereiche, die in em angegeben werden.

#wrapper { width:25em: }



#### ▲ Abbildung 3.9

Elastic Layout bei 100% der initialen Schriftgröße des Layouts aus dem Workshop »Sprechblasen und Teaserboxen« in Kapitel 6

Wie die Abbildungen 3.9 bis 3.12 verdeutlichen, schrumpft und wächst das Design eines solchen Layouttyps mit der Definition für die Schriftgröße des body-Elements. Bei einer Erhöhung dieser Schriftgröße von 100% auf 115% verändert sich somit nicht nur die Schriftgröße der Textinhalte, sondern auch die Maße der mit em gekennzeichneten Elemente dieses Layouts.

body { font-size:115%; }

Weil alle vorhandenen Breiten-, Höhen- und Abstandsangaben mit 115% denselben Ursprung für ihre Maße besitzen, verändern sich auch die Proportionen der Elemente nicht: Die drei Infoboxen mit all ihren Eigenschaften (Breite, Schriftgröße, »runde Ecken«...) samt Illustration wachsen. Voraussetzung für die Skalierbarkeit eines Bildes wie in Abbildung 3.11 ist, dass sich die Bilddatei der Hunde-Illustration im HTMI-Dokument befindet und ihre Maße (Höhe und Breite) ebenfalls in em angegeben werden.



#### ▲ Abbildung 3.10

Elastic Layout bei 115 % der initialen Schriftgröße des Layouts

Reduzieren Sie nun beispielsweise die initiale Schriftgröße für das body-Element auf 85%, werden auch die drei Infoboxen kleiner.

body { font-size:85%; }









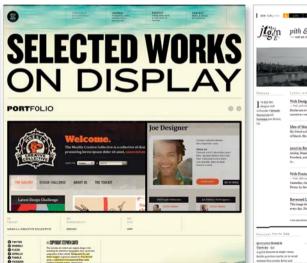
#### ▲ Abbildung 3.11

Verhalten eines Webseitenelements basierend auf der Maßeinheit em bei einer initialen Schriftgröße von 115% (oben), mit dem Originalwert von 100% (Mitte) und einer Schriftgröße von 85% (unten)

#### **◆** Abbildung 3.12

Elastic Layout bei 85 % der initialen Schriftgröße des Layouts







#### Nachteil des elastischen Layouts

Elastische Layouts können diffizil bei Umsetzung eines punktuellen und pixelgenauen Layouts sein, da sich die zu verwendenden Werte der Elemente vererben und somit auch Rundungsfehler auftreten können, die einen größeren Testund Rechenaufwand zur Folge haben als bei fixen Layouts.

#### ▲ Abbildung 3.13

Webseiten mit elastic Layout – von links nach rechts: Pearsonified (pearsonified.com), StephenCaver (stephencaver.com) und JonTangerine (jontangerine.com)

Wenn Sie es konsequent machen, dann können die Proportionen einer solchen Webseite dieses Layouttyps 1: 1 beibehalten werden. Allerdings ist das gerade beim Einsatz von em mit einem nicht unerheblichen Rechenaufwand verbunden. Wenn Sie diesen Aufwand nicht scheuen, können Sie wie in Abbildung 3.11 auch Elemente »punktgenau« ausrichten, sodass diese selbst einer Vergrößerung oder Verkleinerung standhalten und keine ungewollten Abstände oder fehlerhaften Einrückungen entstehen.

### Mischformen

Die Mischform einzelner Layouttypen kann in allen Browsern selbst dann skaliert werden, wenn die Funktion des Webseitenzooms im Browser (beispielsweise IE 6) noch nicht integriert ist. Zudem ragt eine Webseite bei enormer Vergrößerung horizontal nicht über den Rand hinaus und erzeugt somit auch keinen Scrollbalken (Ausnahmen bilden der IE 6 und alle anderen älteren Browser, die mit der Eigenschaft max-width nicht umgehen können).

# 3.4 Mischformen

Da alle drei bisher vorgestellten Layouttypen nicht frei von Nachteilen sind, ist im Lauf der Zeit daraus ein vierter Layouttyp entstanden. Da dieser noch keine offizielle Bezeichnung besitzt, verwende ich dafür den Begriff »elastisches Layout mit Breitenbegrenzung«. Kurz gesagt, dieser Typ könnte sich wie ein elastisches Layout verhalten, nur dass er mit der Eigenschaft maxwidth eine Begrenzung für seine Breite erhält, die mit einem prozentualen Wert versehen ist. Diejenigen Browser (alle aktuell relevanten Browser bis auf den IE 6), die mit der zusätzlichen Eigenschaft zur Breitenbegrenzung umgehen können, verändern somit das Layout wie gewohnt relativ und proportional zur Veränderung der Schriftgröße, allerdings ohne dabei die Breite des Viewports des Browsers zu überschreiten. Die übrigen CSS-Eigenschaften des Layouts basieren unabhängig davon auf dem zuvor vorgestellten elastischen Layout.

# Ein Blogdesign im Retro-Stil

»Retro« als Gestaltungsmittel

Design ist eine ständige und immer wiederkehrende Transformation bestimmter Stile und deren Charakteristika. Auch wenn diese Entwicklung im Webdesign nicht dermaßen verankert ist wie beispielsweise im Pro-



duktdesign, ist Retro-Design entgegen seiner eigentlichen Definition somit immer zeitgemäß. In diesem Workshop wird gezeigt, wie Sie das Retrodesign bei der Umsetzung eines Headers mit Hilfe von CSS3-Eigenschaften umsetzen können.

# Zielsetzungen:

- ► Anlegen eines gestalterischen Elements im Retro-Stil
- ▶ Gestalten der div-Container über lineare und radiale Farbverläufe
- ► Einfügen von Kreisen über Pseudoelemente

**CSS3-Eigenschaften:** linear-gradient, radial-gradient, Pseudoelemente: before und: after, translate...

# Retro im Webdesign

Den Retro-Stil charakterisieren in erster Linie bunte Farben auf dezenten Hintergründen sowie alte vergilbte Abbildungen an- oder eingerissener Fotos, wie beispielsweise die typischen Pin-up-Bilder. Mit diesen den meisten Betrachtern bekannten Stilelementen können Sie eine ganz bestimmte Stimmung vermitteln und dafür sorgen, dass die Webseite im Retro-Design sich von der Masse absetzt und bei den Betrachtern in Erinnerung bleibt.









#### ■ Abbildung 1

Webseiten im Retro-Design (von links oben nach rechts unten): www.style4you.it, www.targetscope.com. www.sottocostoska.it und www.level2d.com

# Farbverlauf per CSS

Dank der CSS3-Eigenschaft gradient mit ihren beiden Arten der Ausrichtung, linear und radial, haben Sie die Möglichkeit, Farbverläufe nicht mehr nur als Grafiken einzubinden, sondern sie über entsprechende CSS-Eigenschaften zu definieren. Der Vorteil ist neben der möglichen Reduzierung von HTTP-Requests je nach Größe der Datei auch die Verringerung des Ladeverhaltens.

Für diesen Effekt benötigen Sie lediglich zwei div-Container innerhalb des Bereiches #retroHeader:

```
<div id="wrapper">
   <div id="retroHeader">
      <div id="headerMotivLeft"></div>
      <div id="headerMotivRight"></div>
   </div>
```





▲ Abbildung 2
Hintergrundgrafik »bg retro.jpg«

Dem body-Element weisen wir als ersten Schritt die Hintergrundgrafik »bg\_retro.jpg« zu. Diese kann sich innerhalb des Hintergrundbereiches unabhängig vom Viewport des Betrachters beliebig oft wiederholen.

```
body {
   margin:0;
  padding:0;
  text-align:center;
  background: url(../images/bg_retro.jpg);
}
```

Den allumfassenden Wrapper mit gleichnamiger ID beschränken wir auf eine Breite von 950 px und richten ihn mittig aus. Dies erreichen wir, indem wir die Werte für den linken und rechten Außenabstand margin auf auto setzen und der Bereich darum, in diesem Fall das body-Element, eine Zentrierung (center) der horizontalen Textausrichtung enthält (text-align).

```
#wrapper {
   width:950px;
   margin:0 auto;
}
```

# 🔵 Lineare Farbverläufe mit Haltepunkten

Da der Retro-Stil in der Regel sehr farbenfroh umgesetzt wird, soll sich dies auch in dem Farbverlauf dieses Workshop-Beispiels widerspiegeln. Dazu weisen wir den beiden im Arbeitsschritt zuvor angelegten Bereichen #headerMotivLeft und #headerMotivRight keinen herkömmlichen Farbverlauf mit einem Start- und Endpunkt zu, sondern einen Farbverlauf mit mehreren »Haltepunkten« (color-stop). Damit ergeben sich innerhalb eines Elements mehrere farblich sichtbare Kontraste.

Da die Positionierung dieser beiden Bereiche absolut ist, sie sich aber nicht innerhalb des Viewports, sondern innerhalb des div-Containers #retroHeader absolut ausrichten sollen, versehen wir diesen div-Container zuvor mit der Eigenschaft einer relativen Position. Alle folgenden Positionierungseigenschaften spielen sich dadurch innerhalb dieses Bereiches ab.

Die Höhe der nun folgenden Elemente beträgt 120 px. Bei vier verschiedenen Farbwechseln stellen wir also jeder Farbe 30 px zur Verfügung. Die einleitende Eigenschaft der einfachen Hintergrundfarbe ist für die Browser gedacht, die die darauffolgenden Eigenschaften des Farbverlaufes nicht umsetzen können. Als Letztes weisen wir den beiden Bereichen, die noch identische Eigenschaften teilen, einen zu 25 % transparenten Schattenwurf zu, der im weiteren Verlauf des Workshops von anderen Elementen aufgegriffen wird.

```
#retroHeader {
  margin: 0 auto:
  position:relative:
  top:0:
  width:100%:
#headerMotivLeft. #headerMotivRight {
  background: linear-gradient(top.
   #b63112. #b63112 25%, #f56e02 25%, #f56e02 50%.
   #f8cc30 50%, #f8cc30 75%, #aeb00c 75%, #aeb00c):
```

Da die soeben festgelegten CSS-Eigenschaften für beide angelegte Bereiche - #headerMotivLeft und #headerMotivRight - gelten, sind diese beiden div-Container, wie in Abbildung 3 zu erkennen, deckungsgleich.



Da beide Elemente allerdings nicht deckungsgleich sein, sondern unterschiedlich ausgerichtet werden sollen, versehen wir den Bereich #header-Motivieft mit dem z-index:1. Dieses Element wird somit die unterste Position aller noch folgenden Elemente einnehmen. Als Nächstes drehen wir dieses Element mit dem CSS3-Transforms-Modul um 4 Grad gegen den Uhrzeigersinn gedreht.

#### ■ Abbildung 3

Darstellung der Farbverläufe der beiden noch deckungsgleichen Elemente

# Unterschiedliche Qualität bei der Kantenglättung

Trotz der interessanten und sehenswerten Möglichkeiten mit CSS3 wie der Neigung und Farbverlaufsgestaltung von Elementen soll an dieser Stelle auch darauf hingewiesen werden, dass eine in solchem Fall notwendige Glättung der Kanten wie in Abbildung 4 nicht von allen Browsern gleich umgesetzt wird. Opera ab Version 11.1, Firefox ab Version 4 und der Internet Explorer ab Version 9 arbeiten bei dieser Eigenschaft wesentlich sauberer als Safari 5 und Google Chrome 13.





## ▲ Abbildung 4 Oben: Der Farbverlauf hat noch

Übergänge und wirkt kantig. Unten: Mit den Übergängen wirkt der Farbverlauf wieder fließend.

```
#headerMotivLeft {
    z-index:1;
    left:0;
    -moz-transform: rotate(-4deg);
    -webkit-transform: rotate(-4deg);
    -o-transform: rotate(-4deg);
    transform: rotate(-4deg);
}
```

Das Problem, das sich aus den bisherigen CSS-Angaben und der Neigung dieses Elements ergibt, wird im oberen Teil von Abbildung 4 sichtbar: Die Farbübergänge sind aufgrund des abrupten Farbwechsels wie beispielsweise bei 25% verzogen und wirken daher an diesen Stellen kantig und unsauber. Es ist also notwendig, diese Übergänge fließender zu gestalten. Dies erreichen Sie, indem Sie den Farbwechsel, wie im folgenden modifizierten CSS-Code, bereits jeweils 2% früher einleiten (23%, 48%, 73% bzw. 98%). Das führt zum Beispiel beim ersten Farbwechsel dazu, dass der Rot-Ton #b63112 von 0 bis 23% »verläuft« und ab diesem Haltepunkt dann ein »echter Farbverlauf« hin zum Orange #f56e02 vollzogen wird, das erst bei 25% startet. Im Bereich von 23% bis 25% werden also Rot und Orange »gemischt«. Dadurch wirkt der Farbwechsel, wie im unteren Teil von Abbildung 4 zu erkennen ist, weniger kantig und stattdessen gleichmäßiger.

Je nach Höhe des Elements, auf das Sie einen solchen Farbwechsel anwenden, müssen Sie diesen »Puffer« zur Umsetzung eines gleichmäßigen Farbüberganges anpassen, denn was 2% bei einem 500 px hohen Element ausmachen, ist bei einem 50 px hohem Element mit denselben CSS-Eigenschaften für den Farbverlauf kaum wahrnehmbar. Um diese fehlerhafte Darstellung der Kantenglättung zu umgehen, lassen wir die beschriebenen Änderungen nun wie folgt in den bereits vorhandenen CSS-Code einfließen:

```
#headerMotivLeft, #headerMotivRight {
    position:absolute;
    height:120px;
    width:100%;
    background-color:#b63112;
/* Mozilla (Firefox, Flock, etc.) */
    background:-moz-linear-gradient(top,
    #b63112, #b63112 25%, #f56e02 25%, #f56e02 50%,
    #f8cc30 50%, #f8cc30 75%, #aeb00c 75%, #aeb00c);
/* WebKit alt (Safari, Chrome, etc.) */
    background:-webkit-gradient(linear, left top, left
    bottom, from(#b63112), color-stop(0.22, #b63112),
    color-stop(0.25, #f56e02), color-stop(0.47, #f56e02),
    color-stop(0.5, #f8cc30), color-stop(0.72, #f8cc30),
```

```
color-stop(0.75. \#aeb00c). color-stop(0.97. \#aeb00c).
   to(#aehOOc)).
/* WebKit neu (Safari, Chrome ab Version 11, etc.) */
   background:-webkit-linear-gradient(top. #b63112.
   #b63112 23%, #f56e02 25%, #f56e02 48%, #f8cc30 50%,
   #f8cc30 73%, #aeb00c 75%, #aeb00c 97%, #aeb00c):
/* Opera ab Version 11.1 */
   background: -o-linear-gradient(
   #b63112, #b63112 25%, #f56e02 25%, #f56e02 50%,
   #f8cc30 50%, #f8cc30 75%, #aeb00c 75%, #aeb00c):
/* aktueller W3C working draft */
   background: linear-gradient(top.
   #b63112. #b63112 25%. #f56e02 25%. #f56e02 50%.
   #f8cc30 50%, #f8cc30 75%, #aeb00c 75%, #aeb00c);
   -moz-box-shadow: Opx Opx 10px Opx rgba(0.0.0..75):
   -webkit-box-shadow: Opx Opx 10px Opx rgba(0,0,0,.75);
   box-shadow: Opx Opx 10px Opx rgba(0,0,0,.75):
```

Zusammen mit der Neigung des div-Containers #headerMotivLeft und den nun angepassten Werten für vier saubere Farbwechsel zeigen Browser wie der 5er-Firefox, 5er-Safari, 13er-Google-Chrome und Opera in Version 11.5 folgendes Bild:



#### ▲ Abbildung 5

So sehen die Farbverläufe auch mit einer Neigung gut aus.

Da der linke Bereich des Elements #headerMotivLeft aufgrund der vollzogenen Neigung nun mehr oder weniger in der Luft hängt, ziehen wir den Bereich über die Eigenschaft translate in Richtung der v-Achse um 35 px nach oben.

```
#headerMotivLeft {
  -moz-transform: rotate(-4deg) translate(0, -35px);
  -webkit-transform: rotate(-4deg) translate(0. -35px):
  -o-transform: rotate(-4deg) translate(0, -35px);
  transform: rotate(-4deg) translate(0, -35px);
```

Nun schließt der linke obere Bereich, wie Sie in Abbildung 6 erkennen, nahtlos am oberen Browserfensterrand ab.

#### Verschiedene Schreibweisen

Da alle Browser für die Umsetzung eines linearen Farbverlaufes noch das für sie notwendige Präfix benötigen, müssen Sie an dieser Stelle die Angaben mehrfach wiederholen. Für webkit-basierte Browser müssen Sie das sogar doppelt tun, denn die März 2011 eingeführte neue Schreibweise (also die obere der beiden Webkit-Varianten) kann bisher nur Google Chrome umsetzen. Safari 5 kommt mit der Schreibweise. die sich übrigens an der von Mozilla orientiert, dagegen noch nicht zurecht – erst ab Version 6 soll das der Fall sein. Bis dahin setzen lediglich die aktuelle Safari-Nightly-Builds (noch in der Entwicklung befindliche [Test-]Versionen mit neuen Funktionen) die neue Schreibweise um nightly.webkit.org

#### Lineare Farbverläufe mit Generatoren

Wem das Erstellen solch detaillierter und mit mehreren Haltepunkten versehener Farbverläufe zu umständlich ist, der kann auch hier auf einen der zahlreichen Onlinegeneratoren zurückgreifen, zum Beispiel den Gradient-Generator unter: westciv.com/tools/ gradients.

#### Abbildung 6 ▶

Das linke Headermotiv wurde um 35 px nach oben gezogen.



Den bisher unveränderten div-Container #headerMotivRight weisen wir im nächsten Schritt die Eigenschaft an, sich nicht mehr über die ganze Breite auszudehnen, sondern auf lediglich 50%. Zudem legen wir dieses Flement mit einem z-index von 4 über den Container #headerMotivLeft. und richten es rechtsseitig aus (right:0). Dieser z-index basiert darauf, dass dieses Element insgesamt drei Elemente des gesamten Headerlayouts überlagern soll und somit aufgrund der absoluten Positionierung innerhalb der gesamten Ebenen an der vierten Stelle aufgeführt werden soll.

Um eine Art optisches Gegenstück zum zuvor leicht gegen den Uhrzeigersinn geneigten Container zu erhalten, neigen wir diesen Container um 14 Grad in die entgegengesetzte Richtung (siehe Abbildung 7).

```
#headerMotivRight {
   z-index:4:
  width:50%:
   right:0:
   background-color: #f8cc30:
   -moz-transform: rotate(14deg);
   -webkit-transform: rotate(14deg):
   -o-transform: rotate(14deg):
   transform: rotate(14deg);
```

#### Abbildung 7 ▶

Neigung des nur noch halb so breiten rechten Headermotivs um 14 Grad

#### **Buchtipp zum Thema** »Retrodesign«

Wer mehr zu den Gestaltungskriterien im Retrodesign erfahren möchte, dem sei an dieser Stelle das Buch »Retrodesign Stylelab« empfohlen: retrodesign-stylelab.com.



Aufgrund des nun wesentlich größer gewählten Neigungswinkels für den Bereich #headerMotivRight muss dieser mit der translate-Eigenschaft auch um einen größeren Wert nach oben gezogen werden.

```
#headerMotivRight {
   -moz-transform: rotate(14deg) translate(0, -67px);
   -webkit-transform: rotate(14deg) translate(0, -67px);
   -o-transform: rotate(14deg) translate(0, -67px);
   transform: rotate(14deg) translate(0, -67px);
```



#### ■ Abbildung 8

Auch das rechte Headermotiv wird nach oben gezogen. Der Wert liegt aber mit 67 px wesentlich höher als der des linken Farbverlaufs

## Radiale Farbverläufe mit Haltepunkten der kleine Kreis

Die beiden di v-Container sind nun fertig. Wir können uns also den unterschiedlich großen Kreisen widmen, die sich an verschiedenen Stellen des Lavouts befinden.

Falls Sie schon einmal einen Blick in den HTML-Code auf der Buch-DVD geworfen haben, fragen Sie sich eventuell, wie das gehen soll. Denn dort wird klar, dass nicht mehr Elemente zur Umsetzung dieser Kreise vorhanden sind. Dass Sie für die Gestaltung eines Elements nicht immer ein »echtes Element« innerhalb des HTML-Dokumentes benötigen, ist unter anderem den seit CSS3 einsetzbaren Pseudoelementen : before und :after zu verdanken. Über diese Eigenschaften können Sie Elemente vor und nach einem Element gestalten. Voraussetzung ist, dass diese Eigenschaften einem »echten Element« zugeordnet werden.

In diesem Workshop soll dem Bereich #headerMotivLeft mit der Eigenschaft : before ein weiteres Element hinzufügt werden. Rein optisch gesehen befindet sich dieses Pseudoelement nicht unbedingt »vor« dem div-Bereich. Die Positionierung wird allerdings wieder ausnahmslos durch die CSS-Eigenschaften bestimmt, die wir auch für dieses Pseudoelement festlegen werden.

Für den ersten und kleinsten aller Kreise legen Sie Eigenschaften wie den Durchmesser von 88 px und eine Einrückung von links um 49 % fest. Zudem definieren Sie ebenso wie bei den linearen Farbverläufen an dieser Stelle wieder eine Fallback-Hintergrundfarbe für solche Browser, die die Eigenschaften für den radialen Farbverlauf nicht umsetzen können. Der z-index:3 sorgt dafür, dass dieser kleine Kreis über dem Container #headerMotivLeft liegt, aber unterhalb des Containers #headerMotivRight und somit von diesem überlagert wird.

```
#headerMotivLeft:before {
   z-index:3:
   top:Opx;
   left:49%;
   height:88px;
   width:88px;
   background-color: #b63112;
   -webkit-border-radius: 44px;
```

```
-moz-border-radius: 44px;
border-radius: 44px;
```

#### »content« in Pseudoelementen

Mit den Pseudoelementen: before und: after ist es möglich, über die Eigenschaft content vor bzw. nach einem Element beliebige Inhalte wie Text oder Bilder einzufügen. Aber auch wenn Sie keine Inhalte über diese Eigenschaft einfügen möchten, müssen Sie sie dennoch angeben – denn ansonsten ist es auch nicht möglich, gestalterische Eigenschaften, wie hier einen radialen Farbverlauf, zu vergeben.

Dem Pseudoelement weisen Sie dann, wie Sie in Abbildung 9 sehen, radiale Farbverlaufseigenschaften zu. Außerdem bekommt der Kreis einen Schattenwurf, den wir bereits bei den linearen Farbverläufen verwendet haben. Der Anteil der einzelnen Farben ist hingegen anders als bei den linearen Farbverläufen. Der innerste grüne Bereich soll hier einen größeren Bereich (40 % statt 25 %) umfassen.

```
#headerMotivLeft:before {
   position:absolute:
   content:"":
   background: -moz-radial-gradient(
   circle contain. #aeb00c 38%. #f8cc30 40%. #f8cc30 53%.
   #f56e02 55%, #f56e02 68%, #b63112 70%, #b63112);
   background:-webkit-gradient(
   radial, 44 50%, 8, 44 50%, 44, from(#aeb00c),
   color-stop(0.4. \#f8cc30). color-stop(0.4. \#f8cc30).
   color-stop(0.55, #f56e02), color-stop(0.55, #f56e02).
   color-stop(0.70, \#b63112), color-stop(0.70, \#b63112),
   to(#b63112)):
   background: -webkit-radial-gradient(circle contain,
   #aeb00c 38%, #f8cc30 40%, #f8cc30 53%, #f56e02 55%,
   #f56e02 68%. #b63112 70%, #b63112);
   background: radial-gradient(circle contain, center,
   #aeb00c 38%, #f8cc30 40%, #f8cc30 53%,
   #f56e02 55%, #f56e02 68%, #b63112 70%, #b63112);
   -moz-box-shadow: Opx Opx 10px Opx rgba(0,0,0,...75);
   -webkit-box-shadow: Opx Opx 10px Opx rgba(0.0.0..75);
   box-shadow: Opx Opx 10px Opx rgba(0.0.0..75):
```

## Abbildung 9 ▶

Der erste, kleine Kreis sitzt bereits richtig.



## **Der mittlere Kreis**

Für den zweiten, mittelgroßen Kreis vergeben wir die gleichen Eigenschaften wie für den ersten Kreis. Die einzigen Unterschiede sind die

Position und der Durchmessers des Kreises sowie die Reihenfolge innerhalb der Stapelung der Elemente über den z-index.

```
#headerMotivLeft:after {
   z-index:2:
   top:10px:
   left:54%:
   position:absolute:
   content:"";
  height:140px:
  width:140px:
  background-color: #f8cc30;
   -webkit-border-radius: 70px:
   -moz-border-radius: 70px:
  border-radius: 70px;
```

Damit dieser Kreis die gleichen Farbverlaufseigenschaften erhält, fügen wir den Selektor #headerMotivLeft:after dem Selektor #header-MotivLeft:before hinzu. Dadurch profitieren beide Kreise von etwaigen Änderungen im Verlauf.

```
#headerMotivLeft:before, #headerMotivLeft:after {
   position:absolute;
   content:"":
  background: radial-gradient(circle contain, center,
   #aeb00c 38%, #f8cc30 40%, #f8cc30 53%,
   #f56e02 55%, #f56e02 68%, #b63112 70%, #b63112);
```



#### **◆** Abbildung 10

Durch den unterschiedlichen z-Index und die Schatten der einzelnen Elemente wirkt der Stapel immer realistischer.

# Die beiden großen Kreise

Um die beiden großen Kreise zu gestalten und zu positionieren, gehen Sie genauso vor wie bei den beiden kleineren Kreisen: Dem rechten Container #headerMotivRight weisen Sie über das Pseudoelement :before alle notwendigen CSS-Eigenschaften zu. Der einzige Unterschied sind die Position und der wesentlich größere Durchmesser des Kreises, der zur »Abdeckung« des rechten Endes dieses div-Containers. #headerMotiv-Right notwendig ist.

Natürlich spielt auch hier der z-index eine wichtige Rolle. Da sich dieser Kreis über allen anderen Elementen befinden soll, ist nun ein Wert notwendig, der den des div-Containers #headerMotivRight übersteigt.

```
#headerMotivRight:before {
   z-index:5;
   top:-85px;
   right:-12%;
   height:220px;
   width:220px;
   background-color:#f56e02;
   -webkit-border-radius: 110px;
   -moz-border-radius: 110px;
   border-radius: 110px;
}
```

#### Zum Nachlesen: Selektoren

Wer sich bei Selektoren und ihren Auswirkungen nicht ganz sicher ist, dem sei Abschnitt 2.3.1, »Aufbau von CSS-Regeln«, empfohlen. Da sich auch dieser Kreis die Eigenschaften des Farbverlaufes mit den anderen beiden Kreisen teilen soll, fügen wir den Selektor #header-MotivRight:before den beiden anderen Selektoren des linken Bereiches des Headermotivs hinzu.

Lassen Sie sich also nicht verwirren: Abbildung 11 zeigt noch einmal deutlich, dass die letztendliche Position des Pseudoelements im Layout nicht der Bezeichnung des Pseudoelements entsprechen muss. So sieht auch das eben erstellte Pseudoelement rein optisch eher wie :after aus, es ist aber über die Eigenschaft :before definiert. Es ist also alles nur eine Frage der Vergabe der entsprechenden Positionierungseigenschaften.

#### Abbildung 11 ▶

Der große Kreis rechts wurde über das Pseudoelement :before des rechten Headerbereiches realisiert.



Die Maße des vierten und letzten Kreises (220 px) sind identisch mit denen des soeben ganz rechts im Headerbereich positionierten dritten Kreises:

```
#headerMotivRight:before. #headerMotivRight:after {
   z-index.6.
   top:-85px;
   right:-12%:
  height:220px:
  width:220px:
   -webkit-border-radius: 110px:
   -moz-border-radius: 110px:
  border-radius: 110px:
```

Der Unterschied ist natürlich die Position ganz links. Wer allerdings einen Blick auf Abbildung 12 wirft, wird sich unter Umständen wundern, warum die Positionierungswerte top:200px und left:-115% genutzt werden muss. Der Grund hierfür ist, dass der Ausgangspunkt des Pseudoelements (0,0) außerhalb des sichtbaren Bereiches liegt. Denn der linke Bereich des div-Containers #headerMotivRight, der für die Position von headerMotivRight:after ausschlaggebend ist, befindet sich aufgrund der Neigung von 14 Grad nicht mehr am ursprünglichen Platz.

```
#headerMotivRight:after {
   top:200px:
  left:-115%;
  background-color:#f8cc30;
}
```

Da sich auch dieser vierte Kreis die Eigenschaften des Farbverlaufes mit den anderen drei Kreisen teilen soll, fügen wir den Selektor #headerMotivRight:after den anderen Selektoren hinzu.

```
#headerMotivLeft:before, #headerMotivLeft:after,
#headerMotivRight:before, #headerMotivRight:after {
   position:absolute:
  content:"":
   background: radial-gradient(circle contain, center,
  #aeb00c 38%, #f8cc30 40%, #f8cc30 53%,
   #f56e02 55%, #f56e02 68%, #b63112 70%, #b63112);
```



#### **◆** Abbildung 12

Der letzte Kreis ist über allen Headerbereichen und Kreisen ausgerichtet und basiert auf dem Pseudoelement : after des rechten Headerbereiches.

# »retro\_pinupTeaser.png«







# ▲ Abbildung 13 Webseiten mit Pin-un-C

Webseiten mit Pin-up-Girls als Eyecatcher: casino-lemonade.com (oben), sparkysgarage.com (Mitte) und lanalandis.com (unten)

# Pin-up-Girl als Eyecatcher

Wie Sie auch an den zu Beginn dieses Workshops vorgestellten Webseiten erkennen, sind Pin-up-Girls ein gern verwendeter Eyecatcher und ein fast unverzichtbares Merkmal von Webseiten im Retro-Design.

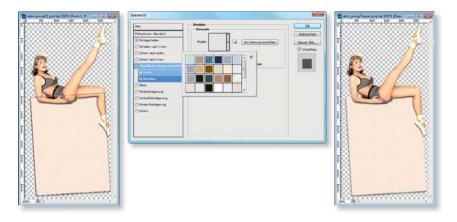
Um die Fotografie, die Sie im linken Teil von Abbildung 14 sehen, als Designelement im Retro-Stil nutzen zu können, stellen Sie das Motiv zunächst in Adobe Photoshop mit dem Lasso frei (mittlerer Teil von Abbildung 14). Außerdem setzen Sie das Pin-up-Girl auf einen leicht geneigten Kasten, in dem Sie beispielsweise Kontaktmöglichkeiten unterbringen können.



#### ▲ Abbildung 14

Arbeitsschritte zur Erstellung der Grafik »retro\_pinupTeaser.png«: Original (links), Freistellen (Mitte), Hinzufügen der Grafik (rechts)

In einem weiteren Arbeitsschritt fügen Sie dem Kasten über den Ebenenstil eine Textur und einen Schatten hinzu.



#### Abbildung 15 ▶

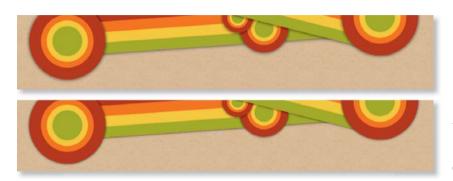
Der Kasten, auf dem das Pin-up-Girl sitzt, bekommt eine Textur und einen Schatten. Die so erstellte Grafik »retro pinupTeaser.png« könnte dann wie in Abbildung 16 im Gesamtlayout ausgerichtet werden.



■ Abbildung 16 Gesamtlayout des Retrodesigns mit Pin-up-Girl

# Darstellung in aktuellen Browsern

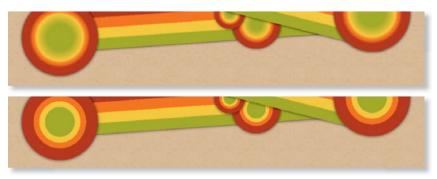
Aufgrund der teilweise noch recht brandneuen Implementierung der Eigenschaften des linearen Farbverlaufes wie beim Opera-Browser in Version 11.5 und der veränderten Schreibweise für Farbverläufe bei webkit-basierten Browsern wie Safari und Google Chrome fragen Sie sich vielleicht, wie die oben beschriebenen CSS3-Eigenschaften von den aktuellen Browsern umgesetzt werden. Abbildung 17 zeigt daher Screenshots der zwei Browser, die die beiden Verlaufseigenschaften (unabhängig von der Schreibweise) korrekt umsetzen. Dies sind der Mozilla-Browser Firefox ab Version 3.6 und Google Chrome ab Version 10.



**◆** Abbildung 17 Darstellung des Retro-Headers in Firefox (oben) und Google Chrome (unten)

Beim Safari-Browser gibt es bei der Umsetzung radialer Farbverläufe einige Abweichungen. Diesem Browser gelingt es nicht, eine scharfe Kante an den Übergängen der Farben zu erzeugen. Selbst der in Arbeitsschritt 3 angelegte Bereich für den Farbübergang von 2% ändert an dieser Stelle nichts

Ein Vergleich der aktuellen 5er-Version des Safari-Browsers mit einem aktuellen Nightly Build (r83429) zeigt hier schon deutliche Unterschiede. Sie sehen also, dass die Browserhersteller aktuell an diesen Eigenschaften arbeiten



#### Radial Gradient für Opera

Wer den Einsatz von JavaScript zur Umsetzung des Retro-Headers für den Opera-Browser vermeiden möchte, der kann in Ausnahmefällen über folgenden CSS-Hack speziell diesem Browser die dafür notwendige Hintergrundgrafik zuweisen: @media all and (-webkitmin-device-pixel-ratio:10000), not all and (-webkit-min-device-pixel-ratio:0) { #retroHeader{ background: url(../images/retro\_header. png;} }

#### ▲ Abbildung 18

Darstellung des Retro-Designs im 5er-Safari (oben) und im aktuellen Nightly Build (unten)

Auch wenn der Unterschied zwischen einem linearen und einem radialen Farbverlauf marginal anmutet, sehen Sie in Abbildung 19, dass der Opera-Browser nur den linearen Verlauf bewältigt. In den Kreisen hingegen stellt er die Fallback-Variante (den jeweils einfarbigen Hintergrund) dar.



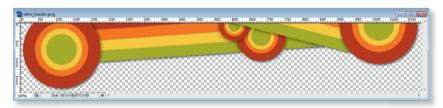
#### ▲ Abbildung 19

Darstellung des Retro-Designs im Opera 11.5

Da selbst die aktuellste Version 9 des Internet Explorers keine der beiden Farbverlaufseigenschaften umsetzen kann, weisen wir allen Versionen dieses Browsers über eine via Conditional Comments referenzierte separate CSS-Datei für den #retroHeader eine Hintergrundgrafik »retro\_header. png« zu (siehe Abbildung 20). Der Nachteil dieses Ansatzes wird dann deutlich, wenn Sie sich die Dateigrößen der Ressourcen ansehen, die bei der Umsetzung dieses Retro-Headers notwendig sind. Bei Chrome, Safari



und Firefox sind dies lediglich 7 KB und ein HTTP-Request für die Styledatei. Beim IE sind dies ebenfalls diese 7 KB, außerdem 1 KB für die Stylesheet-Datei, die für diese IE-Versionen notwendig wird, um dem Bereich #retroHeader die Hintergrundgrafik zuzuweisen, sowie die Grafik selbst mit knapp 100 KB. Wenn Sie nicht auf JavaScript-Ansätze zurückgreifen möchten. Ihre IE-Nutzer aber dennoch etwas von Ihrem Retrodesign sehen sollen, ist diese Vorgehensweise unverzichtbar.



# ▲ Abbildung 20

Hintergrundgrafik »retro\_header.png«



Dieser Workshop hat auf eindrucksvolle Weise gezeigt, in welche Richtung es bei der Gestaltung von Webseiten auf Basis von CSS3-Eigenschaften gehen kann. Auch wenn die Entwicklung durch die Browserhersteller ein stetiger Prozess ist, machen insbesondere die webkit-basierten Browser Google Chrome und Safari sowie der Mozilla-basierte Browser Firefox und dessen »Ableger« deutlich, dass bei entsprechenden Ideen und Know-how kaum Grenzen gesetzt sind.

# Farhverlauf für alle Browser

Wer auf die Verwendung von Grafiken zur Realisierung von Farbverläufen verzichten, dennoch identische Ergebnisse in allen gängigen Browsern erzielen möchte, dem sei das folgende kleine, frei zur Verfügung stehende Script empfohlen: 7synth.com/dev/ gradients.

#### **◆** Abbildung 21

So wird die Hintergrundgrafik im Internet Explorer 9 dargestellt.

# Responsive Webdesign mit Media Queries

Eine Website für mobile Endgeräte optimieren

Mit Media Queries machen Sie die Darstellung, Menge und Reihenfolge der angezeigten Elemente einer Website von der Browserfensterbreite des Geräts abhängig, auf dem die Website aufgerufen wird. Sie können

also unterschiedliche Layouts generieren, ohne dabei auf JavaScript zurückgreifen zu müssen.

Media Types, mit denen Sie Eigenschaften für spezielle Ausgabegeräte (screen, print...) erstellen können, dürften sicher die meisten von Ihnen kennen, aber wie funktionieren Media Queries? Um die Möglichkeiten von Media Queries aufzuzeigen, soll ein bestehendes Projekt genutzt werden. Daher bildet der Workshop »Übersichtlich gegliedert: Navigationen gestalten« aus Kapitel 5 die strukturelle und gestalterische Grundlage dieses Workshops.



## Zielsetzungen:

- ► Gestaltung der Webanwendung dem zur Verfügung stehenden Viewport mit Media Queries anpassen
- ► Definieren von Regeln und weiteren Eigenschaften wie der Browserfensterbreite auf Basis unterschiedlicher Media Types
- ▶ Neuausrichten und Gestalten der Inhalte

CSS3-Eigenschaften: border-radius, Media Queries

Browser-Support von

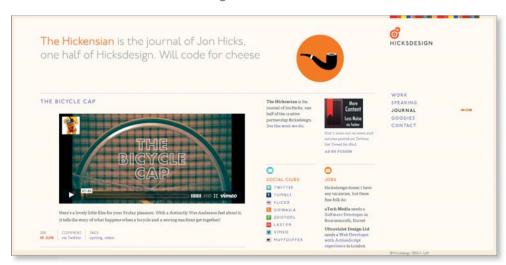
Media Queries

Opera kann seit Version 8 mit Media Queries umgehen, Firefox seit Version 3.5 (Gecko 1.9.1). Safari bietet einen umfangreichen Media-Queries-Support, den Webautoren nutzen können, um mobile Geräte wie iPhones und iPads anzusprechen. Safaris Rendering-Engine-Webkit kommt außerdem in Googles Chrome zum Einsatz, daher ist die Media-Queries-Unterstützung in diesem Browser nahezu identisch mit Safari. Auch der IE ab der aktuellen Version 9 beherrscht Media Oueries. http://www.w3.org/TR/ css3-mediaqueries

## **Funktionsweise von Media Queries**

Bevor die praktische Umsetzung beschrieben wird, sollte einleitend die Funktionsweise von Media Oueries vorgestellt werden. Dazu bieten die bereits erwähnten Media Types einen guten Einstiegspunkt. Wenn Sie bereits separate Stylesheets für die Druckausgabe erstellt haben, kennen Sie schon das Konzept, nach dem bestimmte Stylesheets unter bestimmten Bedingungen aufgerufen werden. Die aus CSS2 stammenden Media Types ermöglichen es also, die Ausgabe von Stylesheets bestimmten Medientypen wie einem Drucker (print), einem Braillegerät (braille), einem TV-Gerät (tv) oder einem Handheld (handheld) zuzuordnen.

Die hier im weiteren Verlauf beschriebenen Media Queries entstammen. CSS3: sie greifen dieses Konzept auf und bauen es weiter aus. Das heißt, es wird nicht nur nach dem Ausgabemedium entschieden, sondern anhand von Eigenschaften und Fähigkeiten von Endgeräten, wie beispielsweise der Breite und Höhe des Viewports etc. Für einen Anwender, der – unabhängig davon, ob er einen Desktop-PC oder ein mobiles Endgerät verwendet einen Browser nutzt, der Media Queries unterstützt, können Sie separate, für das jeweilige Ausgabemedium optimierte Gestaltungseigenschaften festlegen. Voraussetzung für Media Queries ist daher die strikte Trennung von Inhalt und Layout! Ein interessantes Webprojekt, das diesen Aspekt berücksichtigt hat, so dass der Umsetzung mittels Media Queries nichts im Wege stand, ist die Webseite von John Hicks (siehe Abbildung 1).





#### ▲ Abbildung 1

Webseite des Webdesigners John Hicks (hicksdesign.co.uk) in der »Desktopversion« (links) und in der mobilen Version (rechts)

Wie zu erkennen ist, richtet sich das Layout nach der Breite des Ausgabemediums. Wie Sie die dafür notwendigen Regeln definieren, erfahren Sie im weiteren Verlauf dieses Workshops. Um die gewünschten Eigenschaften demonstrieren zu können, wird, wie bereits erwähnt, der Workshop Ȇbersichtlich gegliedert: Navigationen gestalten« aus Kapitel 5 die strukturelle und gestalterische Grundlage dieses Workshops bilden. Das Navigationskonzept wird, ohne dabei den HTML-Code modifizieren zu müssen, grundsätzlich geändert und den Anforderungen der Endgeräte angepasst, von großem bis kleinem Viewport (mobile Endgeräte), so wie es auch bei der Webseite der Webveranstaltung dConstruct aus dem Jahr 2010 geschehen ist (siehe Abbildung 2).





#### ▲ Abbildung 2

Die Webseite der Webveranstaltung dConstruct (2010.dconstruct.org) in der Desktopversion (links) und in der mobilen Version (rechts)

Ziel ist es also, das Layout den Anforderungen eines Nutzers eines mobilen Endgerätes entsprechend zu optimieren.

Anlegen und Kombinieren von Media-Query-Regeln Ähnlich wie bei Media Types gibt es auch bei Media Queries verschiedene Wege, diese zusätzlichen, für bestimmte Endgeräte angelegten Eigenschaften einzubinden. Sie können das 1 ink-Element benutzen, wobei dort das media-Attribut dazu dient, das jeweilige Ausgabemedien für die referenzierte CSS-Datei zu bestimmen:

k rel="stylesheet" type="text/css" href="styles.css" media="screen" />

Bei der Referenzierung über das link-Element können Sie aber auch weitere Eigenschaften als Merkmale miteinander kombinieren:

#### Media Queries im Web

Was alles mit Media Queries möglich ist, zeigt die Webdesign-Galerie der gleichnamigen Seite unter: http://mediagueri.es

```
media="screen and (min-width: 750px)" />
```

Sie können sogar mehrere Media Queries mit den ieweiligen Merkmalen miteinander kombinieren. Dazu trennen Sie die verschiedenen Ausgabemedien durch ein Komma voneinander.

```
<link rel="stylesheet" type="text/css" href="style.css"</pre>
media="screen and (min-width: 750px), projection and
(min-width: 1500px)" />
```

Aus Gründen der besseren Übersicht und des Codeumfangs sollten Sie bei der Vergabe weiterer Eigenschaften für verschiedene Ausgabemedien jeweils separate Stylesheet-Dateien anlegen. Erstellen Sie also besser eine CSS-Datei für Smartphones und eine für Tablets, statt alles in eine Datei zu packen und dann später Eigenschaften doppelt anlegen zu müssen, weil die Gestaltung dann doch noch unterschiedlich ausfallen soll. Unabhängig davon sollten Sie berücksichtigen, dass bei der Verwendung von Media Queries und dafür neu angelegten CSS-Dateien sämtliche Dateien geladen werden, auch die Inhalte der »Desktopwebseiten«, die unter Umständen via Media Queries ausgeblendet worden sind. Bei der Übertragung der vorhandenen Daten führt dies unter Umständen zu unerwünschtem, weil langsamem Ladeverhalten.

Bei wenig zusätzlichem Code ist dies allerdings nicht so sinnvoll. Die notwendigen CSS-Eigenschaften können Sie in einem solchen Fall, so wie auch in diesem Workshop, einfach an das Ende einer bereits vorhandenen CSS-Datei inline anfügen. Das Einfügen muss am Ende geschehen, da die beispielsweise zuvor für große Viewports definierten Eigenschaften von den Eigenschaften für kleine Viewports überschrieben werden müssen.

```
@media screen and (max-width:750px) {
   body { ... }
```

Aber damit nicht genug. Mit der Eigenschaft device-width können Sie auch gezielt die Breite eines mobilen Geräts wie zum Beispiel des iPhones 3G mit einer Bildschirmauflösung von 480×320 Pixel ansprechen:

```
k rel="stylesheet" type="text/css" href="mobile.css"
media="only screen and (max-device-width:480px)" />
```

Da es bei mobilen Endgeräten wie Smartphones und Tablets mit der Portrait- und der Landscape-View zudem zwei unterschiedliche Ansichtsformate zur Darstellung der Inhalte gibt, können Sie mit orientation die Merkmale height und width miteinander vergleichen. Wenn height

#### Den Viewport mit Metaangaben festlegen

Mit folgender Angabe von Metadaten setzen Sie die Breite des Viewports auf die Breite des Displays des Endgerätes: <meta name="viewport"</pre> content="width=device-width" /> Mit folgenden Metadaten wird die Anwendung initial in der Originalgröße angezeigt: <meta name="viewport"</pre> content="initial-scale=1.0" /> Mit folgenden Metadaten wird die Anwendung in 2,5-facher Vergrößerung angezeigt und erlaubt

keine Zoomeigenschaft: <meta name="viewport"</pre> content="initial-scale=2.5,

user-scalable=no" />

größer oder gleich width ist, besitzt die Darstellung den Wert portrait. andernfalls den Wert landscape. Erkannt wird die Ansicht am Gerät übrigens durch einen Bewegungssensor.

Sollte die Gestaltung der Website auf diesen beiden Ansichtsformaten große Unterschiede aufweisen, können Sie iedem Format folgendermaßen eine eigene CSS-Datei zuweisen:

```
<link rel="stylesheet" media="all and (orientation:portrait)"</pre>
href="nortrait.css" />
k rel="stylesheet" media="all and (orientation:landscape)"
href="landscape.css" />
```

Neben der Integration dieser Media-Query-Regeln ist es wichtig, dass Sie dem Viewport-Metatag innerhalb des head-Elements initial weitere Eigenschaften zur optimierten Darstellung der Inhalte zuweisen. Damit können Sie die Darstellung und Skalierung der Inhalte auf Smartphones (beispielsweise iPhone) und Tablets (beispielsweise iPad) gleichermaßen festlegen.

In diesem Workshop wird auf Geräten mit Viewports von über 750 px (z.B. Netbooks, Tablets wie das iPad) die herkömmliche desktop-optimierte Darstellung der Webseite mit der darin enthalten Navigation angezeigt (siehe Abbildung 3).

Es geht also im Folgenden um die Darstellung der Inhalte auf Geräten mit kleineren Viewports. Diesen Geräten weisen wir mittels Media Queries neue CSS-Eigenschaften zu, um eine optimierte Benutzerführung zu erreichen.



#### Media-Query-Abfragen bei großen Viewports

```
iPads (Portrait und Landscape):
  @media only screen and
  (min-device-width:768px) and
  (max-device-width:1024px)
  {...}
iPads (Landscape):
  @media only screen and
  (min-device-width:768px) and
  (max-device-width:1024px)
  and (orientation : land-
  scape) {...}
iPads (Portrait):
  @media only screen and
  (min-device-width:768px) and
  (max-device-width:1024px)
  and (orientation:portrait)
Desktops und Laptops:
  @media only screen and
  (min-width: 1224px){...}
Große Monitore:
  @media only screen and
```

 $(min-width: 1824px)\{...\}$ 

#### ■ Abbildung 3

Breites Ansichtsformat (landscape) auf dem iPad

## Regeln 27			
Bewertungsstern 430   CSS3   Big orange Button = BoB   Bild   box-reflect 437, 438   box-reflect 437, 438   column-count 111   linear-gradient 352   limportant-Regel 23   Bildergalerie 114   linear-gradient 352   linear-gradient 353   linear-gradient 354   linear-gradient 354   linear-gradient 354   linear-gradient 354   linear-gradient 355   linear-gradient 356   linear-gradient 356   linear-gradient 355   linear-gradient 355   linear-gradient 355   linear-gradient 355   linear-gradient 355   linear-gradient 355   linear-gradient 357	Index	Bewertung	Performance 64
Big orange Button — BoB		maschinenlesbare 424	Regeln 27
Bild		Bewertungsstern 430	CSS3
@font-face 99		Big orange Button → BoB	Animation 103, 179
Billograferie 114   Bilnear-gradient 352   Bilmportant-Regel 23   Billograferie 114   Billograferie 114   Billograferie 1152   Billograferie 114   Billograferie 1152   Billograferie 1153   Billograferie 1154   Billograferie 1154   Billograferie 1155   Billogr		Bild	box-reflect 437, 438
Important-Regel 23	@font-face 99	Out-of-the-Box-Effekt 130	column-count 111
Important-Regel 23	@import 25	Bildergalerie 114	linear-gradient 352
A         blockquote 139         Transform 102, 210, 223, 355           abbr-Element 430         Blog         Transition 210, 223           Abkürzungen 430         einspaltiges Design 244         CSS3-Transforms-Modul           A/B-Test 192, 317         Gridlayout 259         Browser-Support 355           Accessibility 15, 315         Kommentardesign 242         rotate 101, 355           address 421         BoB 395         translate 357           Akkordeon-Effekt 362         border-collapse 165         CSS-Datei           Akkordeon-Effekt 362         border-radius         externe 24, 25           Browserunterstützung 374         ohne Präfix 89         Internet Explorer 25           Nachteile 371         box-Shadow 200, 270, 360         CSSPrefixer 258           Alphatransparenz 212         Browser         CSSPrefixer 258           Alphatransparenz 212         Browser         D           Anligharungszeichen         Browser Weiterung         data-transition 392           CSS 142         Operator 412, 416         Debugging 63, 71           Animation 103, 179         Browser-Präfix 29         Default-Stylesheet 22           Einblend-Effekt 293         Button         Button         Dialogfenster           Ankeridentifikator 119         Glow-Effekt 203         Dokumentenfluss 4	!important-Regel 23		transform 123, 205
abbr-Element 430 Abkürzungen 430 Abkürzungen 430 A/B-Test 192, 377 Gridlayout 259 Bos 395 Accessibility 15, 315 Address 421 Bob 395 Akkordeon-Effekt 362 Browser-Interest 252 Akkordeon-Effekt 362 Abkürzung 374 Nachteile 371 Nachteile 372 Nachteile 372 Nachteile 372 Nachteile 372 Nachteile 373 Nachteile 374 Nachteile 375 Nachteile 362 Nachteile 376 Nachteile 376 Nachteile 376 Nachteile 376 Nachteile 376 Nachteile 376 Nachteile 377 Nachteile 377 Nachteile 376 Nachteile 377 Nac		Block-Element 38	CSS3-Modul
Abkürzungen 430  A/B-Test 192, 317  Accessibility 15, 315  Accessibility 15, 315  Accessibility 15, 315  Adobe Kuler 87  Adobe Kuler 87  Adobe Kuler 87  Achoe Kuler 88  Browser-Achoell 41  Verknüpfen 24  Verk	Α	blockquote 139	Transform 102, 210, 223, 355
A/B-Test 192, 317  A/B-Test 192, 317  Accessibility 15, 315  Accessibility 15, 315  Accessibility 15, 315  Adobe Kuler 87  Adobe Kuler 87  Adobe Kuler 87  Akkordeon-Effekt 362  Browserunterstitzung 374  Nachteile 371  Nachteile 371  Nachteile 371  Nachteile 371  Alphakanal 215  Alphatransparenz 212  Alphatransparenz 212  Anführungszeichen  CSS 142  Hintergrundgrafik 143  Animation 103, 179  Einblend-Effekt 293  Ankeridentifikator 119  Ankeridentifikator 119  Ankeridentifikator 119  Assistive Technologien 321  Assistive Technologien 321  Auflapp-Effekt 362  Außanabstand  negativer 107  Aufsalbungszeichen  Colorotate 282  Aussichtung  Rowser 100  Conditional Comments 25, 135, 272  Aussichtung  Nachteile 26  Avatar 255  CSS  Alphatransparenz 212  Browser-Support 355  rotate 101, 355  rotate 101, 355  rotate 101, 355  translate 367  collapse 165  CSS-Datei  externe 24, 25  Internet Explorer 24, 25  Internet	abbr-Element 430	Blog	Transition 210, 223
Accessibility 15, 315     address 421     address 421     Adobe Kuler 87     Adobe Kuler 87     Akkordeon-Effekt 362     Browserunterstützung 374     Nachteile 371     Nachteile 372     Nachteile 373     Nachteile 374     Nachteile 375     Nachte	Abkürzungen 430	einspaltiges Design 244	CSS3-Transforms-Modul
Accessibility 15, 315 address 421 BoB 395 Adobe Kuler 87 Adobe Kuler 87 Akkordeon-Effekt 362 Browserunterstützung 374 Nachteile 371 Nachteile 377 Nachteile 378 Nachteile 378 Nachteile 379 Nachteile	A/B-Test 192, 317		
address 421 Adobe Kuler 87 border-collapse 165 CSS-Datei Akkordeon-Effekt 362 border-radius Browserunterstützung 374 Nachteile 371 Nachteile 371 Nachteile 371 Nachteile 370 Nachteile 370 Nachteile 371 Nachteile 371 Nachteile 370 Nachteile 370 Nachteile 370 Nachteile 371 Nachteile 370 Nachteile 370 Nachteile 370 Nachteile 371 Nachteile 370 Nachteile 371 Nachteile 370 Nachteile 371 Nachteile 370 Nachteile 371 Nachteile 370 Nachteile 26 Nachteile 26 Nachteile 26 Nachteile 26 Nachteile 370 Nachteile 26 Nach	Accessibility 15, 315		
Akkordeon-Effekt 362 Browserunterstützung 374 Nachteile 371 Novereiten 370 Nachteile 370 Novereiten 370 Novereiten 370 Nachteile 371 Novereiten 370 Nachteile 371 Novereiten 370 Nachteile 371 Nachteile 371 Novereiten 370 Nachteile 371 Nachteile 372 Nalphatransparenz 212 Narknien 372 Nalphatransparenz 212 Nalphatransparenz 212 Nalphatransparenz 212 Nalphatransparenz 212 Nachtein 370 Nalphatransparenz 212 Nachtein 370 Nachteile 371 Nachteile 372 Nachteile 372 Nachteile 372 Nachteile 373 Nachteile 374 Nachteile 374 Nachteile 375 Nachteile 375 Nachteile 375 Nachteile 375 Nachteile 375 Nachteile 376 Nac		_	translate 357
Browserunterstützung 374 Nachteile 371 Box-Modell 41 verknüpfen 24 vorbereiten 370 box-shadow 200, 270, 360 CSSPrefixer 258 Alphatransparenz 212 Browser Analyse 71 Anführungszeichen CSS 142 Debugging 63, 71 Hintergrundgrafik 143 Alphatranion 103, 179 Einblend-Effekt 293 Ankeridentifikator 119 Ankerindikator 119, 369 ASCII 403 CS Assistive Technologien 321 Aufklapp-Effekt 362 Außenabstand per CSS gestalten 253 Außenabstand per CSS gestalten 253 Austeile 47 Austeileng 100 Avatar 255 CSS  Internet Explorer 25 verknüpfen 24 verknüpfen 25 verknüpfen 24 verknüpfen 25 verk	Adobe Kuler 87	border-collapse 165	CSS-Datei
Browserunterstützung 374 Nachteile 371 Box-Modell 41 verknüpfen 24 vorbereiten 370 box-shadow 200, 270, 360 CSSPrefixer 258 Alphatransparenz 212 Browser Analyse 71 Anführungszeichen CSS 142 Debugging 63, 71 Hintergrundgrafik 143 Alphatranion 103, 179 Einblend-Effekt 293 Ankeridentifikator 119 Ankerindikator 119, 369 ASCII 403 CS Assistive Technologien 321 Aufklapp-Effekt 362 Außenabstand per CSS gestalten 253 Außenabstand per CSS gestalten 253 Austeile 47 Austeileng 100 Avatar 255 CSS  Internet Explorer 25 verknüpfen 24 verknüpfen 25 verknüpfen 24 verknüpfen 25 verk	Akkordeon-Effekt 362	•	externe 24, 25
Nachteile 371 box-Modell 41 verknüpfen 24 vorbereiten 370 box-shadow 200, 270, 360 CSSPrefixer 258 Alphakanal 215 Breadcrumb-Navigation 300 CTA 395 Alphatransparenz 212 Browser Analyse 71 Statuszeile 410 D Anführungszeichen Browsererweiterung data-transition 392 CSS 142 Operator 412, 416 Debugging 63, 71 Hintergrundgrafik 143 Tails Export 409, 410 Browser-Extensions 72 Animation 103, 179 Browser-Präfix 29 Default-Stylesheet 22 Einblend-Effekt 293 Buchstabenabstand 129 Deklaration 29 Keyframe 179 Button Dialogfenster Ankeridentifikator 119 Glow-Effekt 203 gestalten 137 Ankerindikator 119, 369 ASCII 403 C Assistive Technologien 321 Call-to-Action 395 Dragonfly 75, 76 Ästhetik 15 caption-side 166 Drop Cap 145 Attributselektor 197 cite 141 Drucklayout 33 Aufkalpp-Effekt 362 Außenabstand Clearing 47 E negativer 107 ClearType 99 Einblend-Effekt 123, 331 Aufzählungszeichen Colorotate 282 Element per CSS gestalten 253 column-count 111, 146 ausrichten 210 Ausgabemedium 24 Conditional Comments 25, 135, 272 Ausgabemedium 24 Conditional Comments 25, 135, 272 Austat 255 CSS		ohne Präfix 89	Internet Explorer 25
Alphakanal 215 Alphatransparenz 212 Browser Analyse 71 Anführungszeichen Browsererweiterung CSS 142 Operator 412, 416 Hintergrundgrafik 143 Animation 103, 179 Browser-Präfix 29 Einblend-Effekt 293 Ankerindikator 119 Ankerindikator 119 Assistive Technologien 321 Asthributselektor 197 Aufklapp-Effekt 362 Außenabstand Clearing 47 Aufkalbungszeichen Parker Ausgabemedium 24 Ausrichtung Auschiede 370 Auschtung Auschiede 370	Nachteile 371	Box-Modell 41	
Alphakanal 215	vorbereiten 370	box-shadow 200, 270, 360	CSSPrefixer 258
Alphatransparenz 212 Analyse 71 Anführungszeichen Browsererweiterung CSS 142 Operator 412, 416 Hintergrundgrafik 143 Animation 103, 179 Einblend-Effekt 293 Ankerindikator 119 Ankeridentifikator 119, 369 ASCII 403 Assistive Technologien 321 Attributselektor 197 Aufklapp-Effekt 362 Außenabstand negativer 107 ClearType 99 Auszabenedium 24 Auszichtung Float 47 Auszichtung Float 47 Auszichtung Float 47 Auszichtung Float 47 Austeile 26 Avatar 255 CSS  Browser Präkt 410 Debugging 63, 71 Browser-Extensions 72 Debugging 63, 71 Browser-Extensions 72 Debugging 63, 71 Browser-Extensions 72 Default-Stylesheet 22 Default-Stylesheet 22 Default-Stylesheet 22 Deklaration 29	Alphakanal 215		CTA 395
Analyse 71  Anführungszeichen Browsererweiterung CSS 142 Operator 412, 416 Hintergrundgrafik 143 Animation 103, 179 Browser-Präfix 29 Einblend-Effekt 293 Ankeridentifikator 119 Ankeridentifikator 119, 369 ASCII 403 ASSistive Technologien 321 Call-to-Action 395 Attributselektor 197 Aufklapp-Effekt 362 Außenabstand negativer 107 ClearType 99 Aufzählungszeichen per CSS gestalten 253 Ausgabemedium 24 Ausrichtung Nachteile 26 Avatar 255  Browser-Extensions 72 Debugging 63, 71 Browser-Extensions 72 Default-Stylesheet 22 Deklaration 29 Deklaration	Alphatransparenz 212	_	
Anführungszeichen  CSS 142 Operator 412, 416 Debugging 63, 71  Hintergrundgrafik 143 Tails Export 409, 410 Browser-Extensions 72  Animation 103, 179 Browser-Präfix 29 Default-Stylesheet 22  Einblend-Effekt 293 Buchstabenabstand 129 Deklaration 29 Ekyframe 179 Button Olialogfenster Ankeridentifikator 119 Ankerindikator 119, 369 ASCII 403 C C C Dokumentenfluss 43 Assistive Technologien 321 Astitibutselektor 197 Aufklapp-Effekt 362 Aufklapp-Effekt 362 Aufklapp-Effekt 362 Aufklapp-Effekt 362 Aufklapp-Effekt 362 Aufklapp-Effekt 362 Aufwabstand Clearing 47 Regativer 107 ClearType 99 Einblend-Effekt 123, 331 Aufklahungszeichen Per CSS gestalten 253 Column-count 111, 146 Ausblend-Effekt 370 Ausblend-Effekt 370 Auschleile 26 Avatar 255 CSS  Browser-Präfix 419 Browser-Extensions 72 Default-Stylesheet 22 Deklaration 29 Deklaration		Statuszeile 410	D
CSS 142  Hintergrundgrafik 143  Tails Export 409, 410  Browser-Extensions 72  Default-Stylesheet 22  Einblend-Effekt 293  Reyframe 179  Button  Ankeridentifikator 119  Ankerindikator 119, 369  ASCII 403  Assistive Technologien 321  Attributselektor 197  Aufklapp-Effekt 362  Außenabstand  Regativer 107  Aufkählungszeichen  per CSS gestalten 253  Aussichtung  Auschteile 26  Avatar 255  Deklaration 29  Deklaration	Anführungszeichen	Browsererweiterung	data-transition 392
Hintergrundgrafik 143  Animation 103, 179  Browser-Präfix 29  Buchstabenabstand 129  Keyframe 179  Button  Browser-Extensions 72  Default-Stylesheet 22  Deklaration 29  Deklaration 37  Dekla	_		Debugging 63, 71
Animation 103, 179  Einblend-Effekt 293  Buchstabenabstand 129  Button  Ankeridentifikator 119  Ankerindikator 119, 369  ASCII 403  Assistive Technologien 321  Asthetik 15  Aufklapp-Effekt 362  Außenabstand  Clearing 47  Aufzählungszeichen  per CSS gestalten 253  Ausplend-Effekt 370  Ausplend-Effekt 370  Ausplend-Effekt 370  Avatar 255  Browser-Präfix 29  Buchstabenabstand 129  Button  Dialogfenster  gestalten 137  Dingbats 430  Dokumentenfluss 43  Element  Drucklayout 33  Element  ausrichten 210  in Bewegung versetzen 222  rotieren 225  skalieren 224, 225  Elternelement 35, 60  Avatar 255  CSS	Hintergrundgrafik 143	Tails Export 409, 410	
Keyframe 179ButtonDialogfensterAnkeridentifikator 119Glow-Effekt 203gestalten 137Ankerindikator 119, 369Dingbats 430ASCII 403CDokumentenfluss 43Assistive Technologien 321Call-to-Action 395Dragonfly 75, 76Ästhetik 15caption-side 166Drop Cap 145Attributselektor 197cite 141Drucklayout 33Aufklapp-Effekt 362clear 50AußenabstandClearing 47Enegativer 107ClearType 99Einblend-Effekt 123, 331AufzählungszeichenColorotate 282Elementper CSS gestalten 253column-count 111, 146ausrichten 210Ausblend-Effekt 370column-rule 147in Bewegung versetzen 222Ausgabemedium 24Conditional Comments 25, 135, 272rotieren 225AusrichtungNachteile 26skalieren 224, 225float 47Operatoren 26Elternelement 35, 60Avatar 255CSS		Browser-Präfix 29	Default-Stylesheet 22
Ankeridentifikator 119 Ankerindikator 119, 369 ASCII 403 C C Dokumentenfluss 43 Assistive Technologien 321 Call-to-Action 395 Dragonfly 75, 76 Asthetik 15 Attributselektor 197 Aufklapp-Effekt 362 Außenabstand Clearing 47 Aufzählungszeichen Per CSS gestalten 253 Auslend-Effekt 370 Auslend-Effekt 370 Auslend-Effekt 370 Ausgabemedium 24 Ausrichtung Avatar 255 Avatar 255 CSS  Call-to-Action 395 Dokumentenfluss 43	Einblend-Effekt 293	Buchstabenabstand 129	Deklaration 29
Ankerindikator 119, 369  ASCII 403  Call-to-Action 395 Call-to-Action 395 Dragonfly 75, 76 Drop Cap 145 Drucklayout 33  Aufklapp-Effekt 362 Außenabstand Clearing 47 ClearType 99 ClearType 99 Einblend-Effekt 123, 331  Aufzählungszeichen Per CSS gestalten 253 Ausgabemedium 24 Conditional Comments 25, 135, 272 Ausrichtung Avatar 255  Dragonfly 75, 76 Dragonfly 75, 76 Drop Cap 145 Drucklayout 33  E  E  E  in Bewegung versetzen 222  rotieren 225 skalieren 224, 225 Elternelement 35, 60  CSS	Keyframe 179	Button	Dialogfenster
Ankerindikator 119, 369  ASCII 403  Call-to-Action 395 Call-to-Action 395 Dragonfly 75, 76 Drop Cap 145 Drucklayout 33  Aufklapp-Effekt 362 Außenabstand Clearing 47 ClearType 99 ClearType 99 Einblend-Effekt 123, 331  Aufzählungszeichen Per CSS gestalten 253 Ausgabemedium 24 Conditional Comments 25, 135, 272 Ausrichtung Avatar 255  Dragonfly 75, 76 Dragonfly 75, 76 Drop Cap 145 Drucklayout 33  E  E  E  Einblend-Effekt 123, 331  Element  ausrichten 210  in Bewegung versetzen 222  rotieren 225  skalieren 224, 225  Elternelement 35, 60  Avatar 255  CSS	Ankeridentifikator 119	Glow-Effekt 203	gestalten 137
Assistive Technologien 321 Call-to-Action 395 Dragonfly 75, 76 Drop Cap 145 Attributselektor 197 Cite 141 Drucklayout 33 Aufklapp-Effekt 362 Außenabstand Clearing 47 E  negativer 107 ClearType 99 Einblend-Effekt 123, 331 Aufzählungszeichen Per CSS gestalten 253 Column-count 111, 146 Ausblend-Effekt 370 Ausblend-Effekt 370 Conditional Comments 25, 135, 272 Ausrichtung Nachteile 26 Avatar 255 CSS Dragonfly 75, 76 Dragonfly 75, 76 Drop Cap 145 Drucklayout 33  E  E  E  in Bend-Effekt 123, 331 Element  ausrichten 210 in Bewegung versetzen 222  rotieren 225 skalieren 224, 225 Elternelement 35, 60	Ankerindikator 119, 369		_
Ästhetik 15 caption-side 166 Drop Cap 145 Attributselektor 197 cite 141 Drucklayout 33 Aufklapp-Effekt 362 clear 50 Außenabstand Clearing 47 <b>E</b> negativer 107 ClearType 99 Einblend-Effekt 123, 331 Aufzählungszeichen Colorotate 282 Element  per CSS gestalten 253 column-count 111, 146 ausrichten 210 Ausblend-Effekt 370 column-rule 147 in Bewegung versetzen 222 Ausgabemedium 24 Conditional Comments 25, 135, 272 rotieren 225 Ausrichtung Nachteile 26 skalieren 224, 225  float 47 Operatoren 26 Elternelement 35, 60 Avatar 255	ASCII 403	C	Dokumentenfluss 43
Attributselektor 197 cite 141 Drucklayout 33  Aufklapp-Effekt 362 clear 50  Außenabstand Clearing 47 E  negativer 107 ClearType 99 Einblend-Effekt 123, 331  Aufzählungszeichen Colorotate 282 Element  per CSS gestalten 253 column-count 111, 146 ausrichten 210  Ausblend-Effekt 370 column-rule 147 in Bewegung versetzen 222  Ausgabemedium 24 Conditional Comments 25, 135, 272 rotieren 225  Ausrichtung Nachteile 26 skalieren 224, 225  float 47 Operatoren 26 Elternelement 35, 60  Avatar 255	Assistive Technologien 321	Call-to-Action 395	Dragonfly 75, 76
Aufklapp-Effekt 362 clear 50  Außenabstand Clearing 47  negativer 107 ClearType 99 Einblend-Effekt 123, 331  Aufzählungszeichen Colorotate 282 Element  per CSS gestalten 253 column-count 111, 146 ausrichten 210  Ausblend-Effekt 370 column-rule 147 in Bewegung versetzen 222  Ausgabemedium 24 Conditional Comments 25, 135, 272 rotieren 225  Ausrichtung Nachteile 26 skalieren 224, 225  float 47 Operatoren 26 Elternelement 35, 60  Avatar 255 CSS	Ästhetik 15	caption-side 166	Drop Cap 145
Außenabstand  negativer 107  ClearType 99  Einblend-Effekt 123, 331  Aufzählungszeichen  per CSS gestalten 253  Ausblend-Effekt 370  Ausgabemedium 24  Conditional Comments 25, 135, 272  Ausrichtung  Nachteile 26  Avatar 255  CSS  Einblend-Effekt 123, 331  Element  ausrichten 210  in Bewegung versetzen 222  rotieren 225  skalieren 224, 225  Elternelement 35, 60	Attributselektor 197	cite 141	Drucklayout 33
negativer 107 ClearType 99 Einblend-Effekt 123, 331 Aufzählungszeichen per CSS gestalten 253 Column-count 111, 146 Ausblend-Effekt 370 Ausblend-Effekt 370 Column-rule 147 in Bewegung versetzen 222 Ausgabemedium 24 Conditional Comments 25, 135, 272 Ausrichtung Nachteile 26 Skalieren 224, 225 float 47 Operatoren 26 Elternelement 35, 60 Avatar 255 CSS	Aufklapp-Effekt 362	clear 50	
Aufzählungszeichen  per CSS gestalten 253  Column-count 111, 146  Aussichten 210  Ausgabemedium 24  Conditional Comments 25, 135, 272  Ausrichtung  Nachteile 26  Nachteile 26  Avatar 255  CSS  Element  ausrichten 210  in Bewegung versetzen 222  rotieren 225  skalieren 224, 225  Elternelement 35, 60	Außenabstand	Clearing 47	E
per CSS gestalten 253 column-count 111, 146 ausrichten 210 Ausblend-Effekt 370 column-rule 147 in Bewegung versetzen 222 Ausgabemedium 24 Conditional Comments 25, 135, 272 rotieren 225 Ausrichtung Nachteile 26 skalieren 224, 225 float 47 Operatoren 26 Elternelement 35, 60 Avatar 255 CSS	negativer 107	ClearType 99	Einblend-Effekt 123, 331
Ausblend-Effekt 370 column-rule 147 in Bewegung versetzen 222 Ausgabemedium 24 Conditional Comments 25, 135, 272 rotieren 225 Ausrichtung Nachteile 26 skalieren 224, 225 float 47 Operatoren 26 Elternelement 35, 60 Avatar 255 CSS	Aufzählungszeichen	Colorotate 282	Element
Ausgabemedium 24 Conditional Comments 25, 135, 272 rotieren 225 Ausrichtung Nachteile 26 skalieren 224, 225 float 47 Operatoren 26 Elternelement 35, 60 Avatar 255 CSS	per CSS gestalten 253	column-count 111, 146	ausrichten 210
Ausrichtung Nachteile 26 skalieren 224, 225 float 47 Operatoren 26 Elternelement 35, 60 Avatar 255 CSS	Ausblend-Effekt 370	column-rule 147	in Bewegung versetzen 222
float 47 Operatoren 26 Elternelement 35, 60 Avatar 255 CSS	Ausgabemedium 24	Conditional Comments 25, 135, 272	rotieren 225
Avatar 255 CSS	Ausrichtung	Nachteile 26	skalieren 224, 225
_	float 47	Operatoren 26	Elternelement 35, 60
· I · I · · · ·		CSS	
einbinden 23 <b>F</b>		einbinden 23	F
<b>B</b> Einheiten 33 Farbmodell	В	Einheiten 33	Farbmodell
Benutzerfreundlichkeit 15 Expressions 252 FarbmodellHSLA 303	Benutzerfreundlichkeit 15	Expressions 252	FarbmodellHSLA 303
Benutzer-Stylesheet 22 Funktionsweise 22 Farbschema erstellen 87	Benutzer-Stylesheet 22		Farbschema erstellen 87

Farbverlauf 88, 352	Großbuchstaben	K
Generatoren 231	per CSS 129	Kombinator 28
Haltepunkte 176, 228	Grunge-Stil 210	Kommentar gestalten 242
im Internet Explorer 175		Kontaktdaten 402
Merkmale 89	Н	Konversion 18
ohne Grafiken 174	handheld 24	Konversionsrate 18
Opera 437	hasLayout 46	
per JavaScript 241	hCalendar 412	L
radialer 233	hCalendar.dtend 415	Label
Farbverlaufsgenerator 353	hCalendar.dtstart 415	anklickbares 190
Firebug 72	hCalendar.vevent 413	label-Element 319, 322
Flexibilität 60	hCard 401	Landingpage 433
float	hCard-Creator 401	Landscape-View 387
aufheben 50	Header 210	Layout
zuweisen 48	Hexadezimal-Wert 172	elastisches 59
Floating 47	Hintergrund	fixes 55
Footer 104	fixed 338	fließendes 57
am Browserfenster fixieren 104	skalierbarer 133	mehrspaltiges 146
footerStickAlt 105	transparenter 188	Lesbarkeit 129
form-Element 317	Hintergrundfarbe	Zeilenhöhe 349
Formular 38, 182	alternieren 253	Lightbox
Aktivitätsstatus hervorheben 202	mit Alphakanal 367	mit JavaScript 124
Benutzerführung 315	Hintergrundgrafik 186	ohne JavaScript 114
Bestellen-Button 193	transparente 170	linear-gradient 88, 352
Fehlermeldung 334, 335	hReview 425	Liste
Fehlermeldungen darstellen 198, 200	HTML5 389	Positionierung 128
Konversionsrate erhöhen 192	Hyperlink 421	Load Sharing 70
label 185		
method 184	I	M
Pflichtfelder kennzeichnen 196, 320	IETester 78, 408	Maßeinheiten 33
strukturieren 184	if-Abfrage 25	Margin
umfangreiches 315	Image-Replacement 286	negative 107
versenden 184	Initial 145	max-height 267
FOUC-Effekt 64	Inline-Block-Element 41	Media Queries 342, 375
	Inline-Element 38, 39, 326	Browser-Support 376
G	Inline-JavaScript 63	einbinden 377
get 184	Inline-Style 63	Formular 339
Google	Interaktion 210	Funktionsweise 376
Google Font API 99, 110, 139	Internet Explorer 25	Nachteile 385
Google Font Directory 99	Neigung 288	Media Types 24, 376
Mikroformate 423	iPad	Microsoft-Filter 31
Graceful Degradation 52, 54	iPad-Simulator 380	Mikroformate 409
Gradient-Generator 231	Testumgebung 380	hCalendar 412
Grafik	iPhone	hCard 400
wiederholen 108	Testumgebung 385	hReview 424
Gravatar 255	0 0 5 5	Mikroformat.fn 401
Größenangabe	J	Mikroformat.geo 402
relative 33	jQuery 274	Mikroformat.org 401
Gridgenerator 259	jQuery Mobile 388	MicroformatValue Class Pattern 415
Gridlayout 248		Telefonnummer 403, 404

Mindesthöhe 267	Positionierung	Screen Real Estate 363
min-height 267	absolute 44, 128, 173, 210	Screenshot-Services 80
IE 6 351	fixe 45	Search Engine Research Page → SERP
Minify CSS 65	relative 43, 128, 210	Seitenoptimierung 63
Mobile Endgeräte 375	post 184	Selektor 28
Mobile Website 386	Präfixe 29	Semantik 18
Anforderungen 387	Präfix-o- 437	SEO 17
Viewport 387	Präfix-webkit- 437	SEOKeywords 364
Multi-Column-Layout 111	Progressive Enhancement 52, 290	SEOÜberschriften 364, 366
Multi-Column-Layout-Modul 146	Pseudoelement 203	SERP 18, 410
MultipleIE 78	after 271, 305	Shorthand-Property 64
Multi-Safari 79	before 254, 305	Slide-Effekt 362
	content 234	Sliding-Doors-Technik 90
N	Pseudoklasse 370	Spaltenlayout 146
Namenskonvention 31	first-child 306, 307	Trennlinie einfügen 148
Navigation 84	last-child 268, 308, 309	Spezifität 23
Akkordeon-Effekt 363	Pseudoselektor	Spiegelung 438
Breadcrumb 300	nth-child() 306	Sprache 139
mit CSS3 88		Sprechblase 132, 305
mit Grafiken 90	R	Spritegrafik 151, 312
Struktur anlegen 85	Rahmenabstand 165	Sprite-Grafik 69
Navigationspunkt	Raster 248	Stapeleffekt 359
ausrichten 86	Reiternavigation 88	Stapelung 45, 360
mit runden Ecken 88	Responsive Webdesign 385	style-Attribut 23
Negationsselektor 368	RGBA 164	style-Element 23
Nutzbarkeit 15	Google Chrome 284	Suchmaschinencrawler 425
Nutzerzentriertes Design 16	Safari 284	Suchmaschinenoptimierung → SEO
Tratzerzentirertes besign 10	rotate 225	540aseenepae.a8
0	101416 225	Т
Onpage-Optimierung 17, 363	S	Tabelle 158
Opera Opera	scale 224	Beziehungen zwischen Zellen 160
radialer Farbverlauf 240	Schatten 359	border-collapse 165
Operator 26	mit Pseudoelementen 203	caption 160
Out-of-the-Box-Effekt 130	nach innen 200	caption-side 166
out of the box Ellekt 150	Schrift	Grundgerüst anlegen 159
Р	einbinden per Google Font API 99	Kopfzelle 168
Parallax-Effekt 210, 221	Glättung 99	mit Hintergrundgrafiken 167
Performance verbessern 63	Google Font API 110	ohne Hintergrundgrafiken 174
Pflichtfeld 182	Schriftart	scope 161
Pflichtfeld kennzeichnen 196	Georgia 418	Spaltenüberschrift 167
	Helvetica 417	Tabellenstruktur 162
PlayStation 34	• •	
PNG-8 94	Schriftgräße	Tabellenspalte
PNG-24 91	Schriftgröße	optisch hervorheben 158
pointer-events 308	absolute 35	Tab-Navigation 84
Pop-up	berechnen 36	Tails Export 409
Einblend-Effekt 123	em 35	Teaserbox
ohne JavaScript 114	Maßeinheit 33	Out-of-the-Box-Effekt 130
positionieren 121	Schlüsselwörter 37	text-indent 169
Schließen-Funktionalität 122	Screenreader 320	text-shadow 144, 269
Portrait-View 387	Tabellendaten strukturieren 162	Textspalte 111

Transparenz 215 TweakPNG 406

#### U

Überlagerungseffekt 221 Überschrift Keywordposition 364, 366 mit Hintergrundfarbe 97 neigen 102 Underscore-Hack 275, 276 Unicode 141 Unterstützungstechnologien 321 Usability 16, 315 bei Formularen 196 User-Stylesheet 22

#### V

Validierung 71

vCard 401 vCard.email 403 vCard.fn 401 vCard.geo 403 vCard.org 401 vCard.tel 402 Veranstaltungsdaten 412 in Outlook 422 Verknüpfung 185 Vertikaler Rhythmus 148 Visitenkarte 401

#### W

WAI-ARIA 184, 320 WCAG 2.0 318 Web-Performance-Optimierung 63 Webseitenfooter → Footer Weichzeichner-Effekt 144 Whitespace 421

#### Z

Zeilenhöhe 129 Zeilenumbruch 38 z-index 45, 215, 217, 360 Zitat 38, 136 Anführungszeichen 142 strukturieren 138 Zoomfunktion 118