

Bewegungstherapie für Webseiten

Flash und JavaScript bekommen Konkurrenz von ungewohnter Seite: Das altbekannte CSS sorgt mit neuen Spezifikationen für mehr Bewegung auf Webseiten. Mit einer neuen Generation von Browsern können Webworker dadurch ihre Projekte für Besucher nutzbarer und attraktiver machen.

Mit CSS3 eröffnet sich ihnen eine Fülle von Animationsmöglichkeiten, um Webseiten interaktiver zu gestalten, ohne auf Flash oder JavaScript zurückgreifen zu müssen. CSS hat einen entscheidenden Vorteil gegenüber Flash: Das darunterliegende HTML bleibt weitestgehend gleich, also auch suchmaschinen-tauglich und barrierefrei. So können Sie durch bewegte Abläufe Ihre Webseite aufpeppen, aktive Bereiche können auf diese Weise durch subtile Animationen für den Benutzer deutlicher herausgestellt werden.

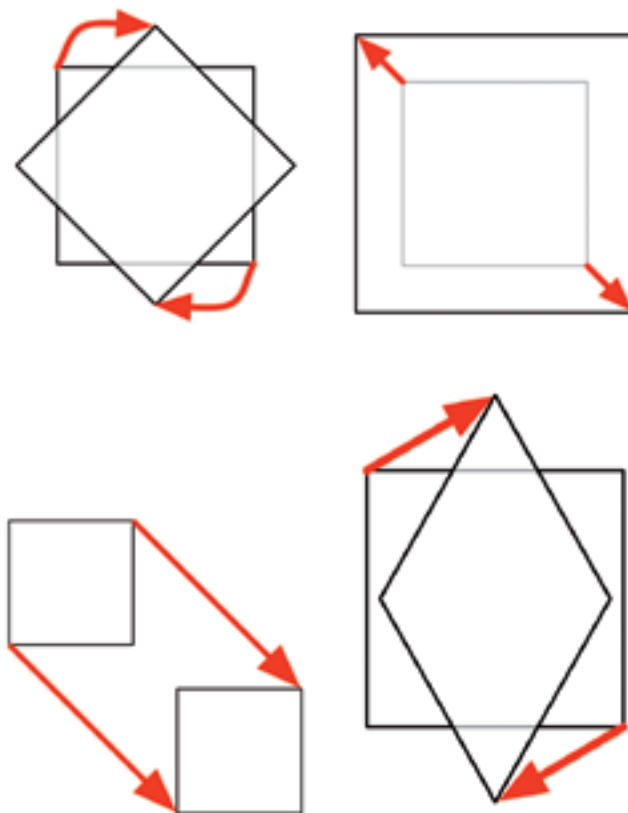
Da die Animationen rein grafischer Natur sind und auf bestehende Funktionen aufgesetzt werden, zeigen ältere Browser automatisch eine weniger angereicherte Form an, *Progressive Enhancement* (dt. Progressive Verbesserung) ¹ hat bei der Entwicklung dieser Standards eine große Rolle gespielt. In Zukunft werden Webnutzer mit noch reichhaltigeren Benutzerschnittstellen (Interfaces) auch auf Webseiten zu tun haben. CSS3 gibt uns Webentwicklern Werkzeuge in die Hand, um diese Ansprüche moderner Webseiten zu erfüllen. Endlich ist es uns möglich, mit reinem CSS dem Benutzer bessere Rückmeldung bei Interaktionen zu geben und mehr zu tun, als nur die Hintergrundfarbe zu ändern. Sie können jetzt eine Aufklappnavigation langsam von oben nach unten aufrollen lassen, ohne eine JavaScript-Bibliothek einzubinden. Die native Einbindung hat neben dem leichteren Seitengewicht noch den entscheidenden Vorteil, dass Browser direkt auf das System zugreifen können, um eine bessere Performance zu bieten.

Das Standards-Triumvirat

Drei Spezifikationen (momentan im Working-Draft-Status) bilden die Grundlagen der neuen Darstellungsmöglichkeiten, die ursprünglich von Apple an das W3C herangebracht wurden: CSS 2D Transforms ², CSS Transitions ³ und CSS Animation ⁴. Die Dokumente ergänzen sich hervorragend und sorgen für Bewegung auch auf Ihrer Webseite.

Transformationen

Um beliebige Elemente frei in der Fläche zu verändern, sind Transformationen gedacht. Durch so genannte Funktionen sorgen sie dafür, dass sich Elemente beliebig drehen, skalieren, verschieben und stauchen lassen. Die Notation erfolgt dabei durch das CSS-Attribut `transform` und die entsprechende Funktion als Funktionswert:

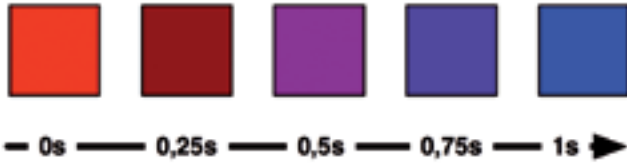


Die Auswirkungen der Dreh-, Skalier-, Verschiebe- und Schrägstellenfunktionen

Durch `transform: rotate(45deg)`; wird das Element um 45° im Uhrzeigersinn gedreht, `transform: scale(1.5)`; skaliert das Element auf 150%. Über die `translate`-Funktion wird das Element verschoben: `transform: translate(100px, 100px)`; . Auch die Möglichkeit, ein Element schrägzustellen, gibt es: `transform: skew(15deg, 15deg)`; stellt das Element um 15° schräg.

Diese Angaben können auch beliebig kombiniert werden, so führt die Angabe `transform: translate(100px, 100px) scale(1.5) rotate(45deg)`; dazu, dass das Element verschoben, skaliert und gedreht wird. Diese Transformationen sind noch statisch und werden nicht animiert. Sie finden zudem im

zweidimensionalen Raum statt. Um Elemente auch in die dritte Dimension zu transformieren, gibt es eine eigene Spezifikation (CSS 3D Transforms [6](#)), deren Umsetzung in den Browsern aber noch extrem dürftig ist.



Transitionen

Eine Transition lässt den Browser geschmeidig zwischen zwei Zuständen eines Elementes über einen Zeitraum wechseln. Das klingt zunächst komplizierter als es in Wirklichkeit ist. Ist die Hintergrundfarbe eines Links beispielsweise Rot und wird beim Überfahren blau, so findet derzeit ein einfaches Umschalten statt, sobald die Maus den Link berührt. Definieren wir nun wie folgt eine Transition, ist der Übergang fließend:

```

CSS
a {
  background: #f00;
  transition: background 1s linear;
}
a:hover {
  background-color: #00f;
}

```

Die Notation bedeutet, dass die Hintergrundfarbe über eine Sekunde linear angepasst wird, also nach einer halben Sekunde ein Lila als Übergangsfarbe zu sehen ist. Nicht alle CSS-Eigenschaften können animiert werden, die Spezifikation hält aber eine Liste der Eigenschaften [7](#) vor, grundsätzlich sind Größenangaben und Farben sowie CSS3-Hintergrund-Gradients und Transformationen animierbar. Um also ein Element um seine Mittelachse zu drehen, können wir folgenden Code benutzen:

```

CSS
div { transition: transform 1s linear; }
div:hover { transform: rotate(360deg); }

```

Neben linear kann als transition-timing-function auch eine dieser Varianten benutzt werden:

ease-in: langsamer Start

ease-out: langsames Ende

ease-in-out: eine Kombination aus ease-in und ease-out

cubic-bezier(x1,y1,x2,y2): Definiert eine Bezierkurve, die das Verhalten der Transition bestimmt.

Why waste time using CSS3 when only minority browsers will see it? – I say „Don't be a lazy fucker!“ (Andy Clarke) [5](#)

Animationen

Das explizite Definieren von Animationen bedeutet für Sie zwar einen wesentlichen Mehraufwand bei der Erstellung, dafür dürfen Sie sich auch auf volle Kontrollmöglichkeiten freuen. Sie können wie in Flash Keyframes erstellen, in denen Sie das genaue Aussehen des Elements festlegen können, zwischen den einzelnen Keyframes wird eine Transition angelegt. Beispiel:

```

CSS
@keyframes "wobble" {
  0% { left: 100px; }
  40% { left: 150px; }
  60% { left: 75px; }
  100% { left: 100px; }
}

```

Die Animation mit dem Namen wobble sorgt dafür, dass beim animierten Element zu Beginn der Animation left auf 100px steht. Nach 40 % der Animationszeit, die bei jedem Element individuell eingestellt werden kann, steht left auf 150px usw. Auch die Art der Transition kann bestimmt werden, indem Sie animation-timing-function auf einen der von den Transitionen bekannten Werte setzen (linear, ease-in, ease-out, ease-in-out, cubic-bezier).

Um einem Element die Animationsabfolge zuzuweisen, wird der animation-name-Eigenschaft der Name zugewiesen: div { animation-name: "wobble"; }, über animation-duration wird die Dauer der Animation bestimmt und über animation-iteration-count die Anzahl an Wiederholungen. Über die animation-direction kann festgelegt werden, wie sich die Animation am Ende verhält, sofern sie wiederholt wird: alternate dreht bei jedem zweiten Durchgang die Animation um. Es ist ebenso möglich, den Beginn der Animation zu verzögern, dazu steht animation-delay zur Verfügung.

```
CSS
div {
  animation-name: "wobble";
  animation-duration: 3s;
  animation-delay: 1s;
  animation-iteration-count: infinite;
}
```

Nachdem die Seite geladen wurde, passiert zuerst nichts, nach einer Sekunde springt das (absolut positionierte) Element auf 100px, bewegt sich dann wie oben beschrieben innerhalb von 3 Sekunden zuerst auf 150px und dann auf 75px, bevor das Element wieder zurück auf 100px kommt, von wo aus die Animation wiederholt wird.

In der Praxis

Um die neuen CSS-Eigenschaften einsetzen zu können, benötigen wir natürlich – abgesehen von den Spezifikationen – auch die Unterstützung in Browsern. Das Webkit-Team ist hier federführend, doch auch Firefox und Opera holen auf und implementieren in den Versionen 3.6 bzw. 10.5 immerhin CSS-Transformationen und -Transitionen.

Wichtig ist, dass sich diese CSS-Eigenschaften noch in der Entwicklung befinden und daher nur mit Herstellervorspann funktionieren: `-webkit-` für Safari, `-moz-` für Firefox und `-o-` für Opera. Aus `transform` wird so `-webkit-transform`, `-moz-transform` und `-o-transform`.

Ältere Modelle der Browser können diese Angaben aber nicht interpretieren. Es ist dann anhand des Projektes zu entscheiden, ob sie vernachlässigt werden können. Der Internet Explorer hingegen kann auch in den aktuellen Versionen mit keiner dieser Spezifikationen etwas anfangen. Paul Bakaus hat zumindest für Transformationen eine Lösung gefunden: Mit den proprietären Filtern des Internet Explorers lassen sich diese Effekte nämlich auch generieren, seine JavaScript-Bibliothek „Transformie“ [8](#) nimmt uns die Arbeit des Umrechnens ab. Nachteil dieser Lösung: ist JavaScript deaktiviert oder ausgefiltert, werden Elemente nicht transformiert.

Einsatzzwecke

Man kann sich sicherlich diverse Effekte vorstellen, die per CSS realisiert werden können, beispielsweise Zoom-Effekte in einer Navigation zu verwenden oder die Möglichkeit, Daten spannend in Szene zu setzen. Im Folgenden einige Beispiele:

Um den beliebten „Yellow Fade“-Effekt zu kreieren, können Sie sich dieser Konstruktion bedienen:

```
CSS
.yellowfade {
  animation: yellowfade 4s;
}
@keyframes ,yellowfade` {
  0% { background-color: #ff9; }
  75% { background-color: #ff9;
        animation-timing-function: ease-out; }
  100% { background-color: #fff; }
```

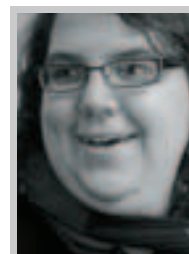
Auch die Interaktion mit einem (z. B. auf einer Lightbox) verlinkten Bild kann verbessert werden:

```
HTML
<a href="bild.jpg" class="thumbnail">
  
</a>
```

```
CSS
.thumbnail {
  display: block;
  float: left;
  width: 100px;
  margin: 5px;
  transition: box-shadow 1s, transform 1s;
}
.thumbnail:focus, .thumbnail:hover {
  box-shadow: 0 0 10px #666;
  transform: scale(1.01);
}
```

Durch das leichte Hochskalieren des Bildes beim Überfahren mit der Maus und dem Erscheinen des Schattens wird der Link sofort fassbar und für den Nutzer klar, dass er damit interagieren kann. Diese und weitere Praxisbeispiele finden Sie laufend aktualisiert in meinem Labor [9](#).

Autor » Links » Quellennachweis



Eric Eggert

Als freier Webentwickler spezialisiert er sich auf sehr gut nutzbare User Interfaces und barrierefreie, standardkonforme Webseiten, bloggt [9](#) und twittert [10](#) darüber. Er gibt sich nicht mit dem Status quo in den Browsern zufrieden, sondern erforscht deren Grenzen und reizt ihre Fähigkeiten aus. Seine Webseite : outlineweb.de

Link-Code [fa6bb1](#)
 Twitter-Hashtag [#wsm0510-45](#)
 Twitter-Account [yaeil](#)