Tag 6 CSS: Transition & Animation

---

1. CSS Transitions

1.1 Was ist eine Transition?

Transitions ermöglichen sanfte Übergänge zwischen zwei Zuständen eines Elements. Wenn z. B. eine `width`, `color` oder `opacity`-Eigenschaft per Hover verändert wird, sorgt eine Transition dafür, dass die Änderung **animiert über eine Dauer hinweg** stattfindet, statt abrupt.

> Transitions benötigen einen **Trigger**, meist ein Pseudozustand wie `:hover`, `:focus` oder Klassenwechsel via JavaScript.

1.2 Einzelne Transition-Properties im Detail

Property	Bedeutung
`transition-property`	Bestimmt, welche CSS-Eigenschaft(en) animiert werden sollen
`transition-duration`	Legt fest, wie lange die Transition dauert (z. B. `0.5s`, `300ms`)
`transition-timing-function`	Gibt an, wie sich die Geschwindigkeit der Animation im Zeitverlauf verändert (z. B. `ease`, `linear`, `ease-in`)
`transition-delay`	Verzögerung, bevor die Animation startet

Beispiel für die getrennte Deklaration:

```css
.box {
  transition-property: width, background-color;
  transition-duration: 0.5s;
  transition-timing-function: ease-in-out;
  transition-delay: 0.2s;
}
```

1.3 Shorthand-Verwendung für Transitions

Alle vier Eigenschaften lassen sich auch zusammenfassen:

```css
transition: background-color 0.3s ease 0.2s;
```

> Diese Kurzschreibweise spart Platz, ist aber weniger flexibel bei komplexen Effekten mit mehreren Eigenschaften.

1.4 Übergangsfunktionen (`timing-function`)

Wert	Beschreibung
`linear`	Gleichmäßige Bewegung
`ease`	Standard, beginnt und endet langsamer
`ease-in`	Langsamer Start, danach schneller
`ease-out`	Schneller Start, langsameres Ende
`ease-in-out`	Langsamer Start und langsames Ende
`cubic-bezier(x1,y1,x2,y2)`	Benutzerdefinierte Kurve

Beispiel mit `cubic-bezier`:

```
transition-timing-function: cubic-bezier(0.42, 0, 0.58, 1);
```

Tools wie cubic-bezier.com helfen beim Experimentieren.

1.5 Beispiel 1: Button-Farbwechsel

```
.button {
  background-color: steelblue;
  color: white;
  padding: 10px 20px;
  transition: background-color 0.3s ease;
}
.button:hover {
  background-color: crimson;
}
```

1.6 Beispiel 2: Größe & Farbe

```
.element {
  width: 100px;
  height: 100px;
  background-color: teal;
  transition: width 0.5s, height 0.5s, background-color 0.5s;
}
.element:hover {
  width: 150px;
  height: 150px;
  background-color: tomato;
}
```

1.8 Transitionfähige Eigenschaften

- `color`
- `background-color`
- `width`, `height`
- `margin`, `padding`
- `opacity`
- `transform`

> Properties wie `display`, `visibility`, `position` können **nicht** animiert werden!

---

2. CSS Animations

2.1 Was ist eine Animation?

Im Gegensatz zur Transition benötigt eine Animation **keinen Trigger** und läuft unabhängig ab. Sie eignet sich für **komplexe, wiederholte oder zeitgesteuerte Bewegungen**, z. B. Laufschriften, Loops, Banner.

2.2 Einzelne Animation-Properties im Überblick

Property	Bedeutung
`animation-name`	Verweist auf die `@keyframes`
`animation-duration`	Dauer der gesamten Animation
`animation-timing-function`	Verlauf der Bewegung
`animation-delay`	Wartezeit vor Start
`animation-iteration-count`	Anzahl Wiederholungen (`1`, `infinite`)
`animation-direction`	Richtung der Animation (`normal`, `reverse`, `alternate`)
`animation-fill-mode`	Verhalten vor/nach der Animation (`none`, `forwards`, etc.)

2.3 Shorthand für `animation`

```
animation: name duration timing delay iteration direction fill-mode;
```

Beispiel:

```
animation: bounce 1s ease 0s infinite alternate both;
```

2.4 Grundaufbau einer CSS-Animation

```
@keyframes anim-name {
  from {
    transform: translateX(0);
  }
  to {
    transform: translateX(100px);
  }
}

.element {
  animation: anim-name 2s ease-in-out 0s infinite alternate forwards;
}
```

2.5 Beispiel: Einfache Slide-Animation

```
@keyframes slide {
  from { transform: translateX(0); }
  to   { transform: translateX(200px); }
}

.box {
```

```
  width: 100px;
  height: 100px;
  background-color: royalblue;
  animation: slide 2s ease-in-out infinite alternate;
}
```

2.6 Beispiel: Farbverlauf mit mehreren Keyframes

```
@keyframes colorPulse {
  0%   { background-color: blue; }
  50%  { background-color: green; }
  100% { background-color: red; }
}

.element {
  animation: colorPulse 3s linear infinite;
}
```

2.7 Animationen kombinieren

Mehrere Animationen können gleichzeitig laufen:

```
.element {
  animation: moveRight 2s ease-in, fadeIn 1s ease-in 1s forwards;
}
```

> Dies ermöglicht **sequentielle Animationen** (z. B. zuerst einblenden, dann bewegen)

---

3. Übungsideen / Projektvorschläge

Übung: Hover-Button

Erstelle einen Button, der bei Hover:

- seine Farbe ändert
- sich leicht vergrößert

Übung: Slide-in beim Laden

Ein Textblock soll bei Seitenaufruf von links einblenden (mit `@keyframes`)

Projektidee: Animiertes Portfolio-Kärtchen

- On Hover: Farbe, Schatten, Schriftgröße verändern (Transition)
- On Load: Kärtchen slidet nach oben und fade-in (Animation)