Animationer

Transitions & keyframes

annika.lindberg@zocom.com

Idag ska vi prata om..

- UX-design och animationer
- Transitions och Keyframes
- Bootcampet

UX-design ❤️ animationer

- Estetiskt vs. användarupplevelse
- Skapa engagement
- Ge information till användaren
- “micro interactions” (visuell feedback)
- Mer naturlig upplevelse
- Vägleder användaren (ökad användbarhet)
- Riktar uppmärksamheten “rätt”
- Underlättar inlärning
- Responsivitet

Inspo 😍

1. Exempel på animerad meny: Trickle
2. Material Design: Transitions, Easing and duration
3. Inspiration: Dribbble
4. NASA använder animationer för att visualisera komplexa vetenskapliga data och göra det tillgängligt och förståeligt för allmänheten:

   Exoplanet Exploration: Planets Beyond our Solar System (nasa.gov)
5. Examples of UI/UX Animation in Website Design

Vad är en animation?

I CSS definieras animationer genom att beskriva **vad som ska förändras** (t.ex. position, färg eller storlek) och **hur förändringen ska ske över tid**.

Animationens byggstenar

För att skapa en animation i CSS behöver vi förstå grunderna, eller "byggstenarna", som är:

- Starttillstånd och sluttillstånd – Vad är elementets ursprungliga tillstånd och vad vill vi att det ska förändras till?
- Tidslinje – Hur lång tid ska förändringen ta, och ska det finnas mellanliggande steg?
- Timing-funktioner – Hur ska förändringen ske? Ska det vara mjukt, linjärt eller accelerera?

Skillnaden mellan övergångar och animationer

Övergångar (transitions):

- En övergång gör att en förändring mellan två tillstånd sker smidigt. T.ex. när du hovrar över en knapp och dess bakgrundsfärg ändras gradvis istället för omedelbart.

- **Exempel:**

```css
button {
    background-color: blue;
    transition: background-color 0.3s ease;
}

button:hover {
    background-color: green;
}
```

Animationer

- Animationer är mer flexibla och kraftfulla. De låter dig definiera flera steg, där du kan skapa komplexa rörelser eller färgskiftningar.
- **Exempel:**

```
@keyframes bounce {
    0% { transform: translateY(0); }
    50% { transform: translateY(-20px); }
    100% { transform: translateY(0); }
}

div {
    animation: bounce 1s infinite;
}
```

Animation och @keyframes

CSS-egenskapen *animation* är en shorthand för flera egenskaper

```css
.box {
  animation-name: example;
  animation-duration: 5s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}
```

```css
.box {
  animation: example 5s linear 2s infinite alternate;
}
```

Timing-funktioner

Timing-funktioner påverkar hastigheten och känslan i en animation. Här är några vanliga:

1. **Ease (standard):**
    - Startar långsamt, accelererar och saktar sedan ned mot slutet.
2. **Linear:**
    - Rörelsen sker i jämn hastighet.
3. **Ease-in:**
    - Startar långsamt och accelererar sedan.
4. **Ease-out:**
    - Startar snabbt och saktar ned mot slutet.
5. **Cubic-bezier:**
    - En anpassad funktion där du kan skapa din egen timing.

Transition exempel

```
.highlight {
    background-color: #F7F8E0;
    transition: background-color 2.5s;
}

.highlight:hover {
    background-color: #F3E2A9;
}
```

Timing functions

linear: ändrar värdet linjärt, dvs med samma hastighet hela animationen.

ease-in: börjar långsamt.

ease-out: saktar in mot slutet.

ease-in-out: långsammare i början och slutet av animeringen.

cubic-bezier: mer avancerad funktion som kan ställa in hastigheten i olika delar av animationen i detalj.

```
transition: color 1s linear;
```

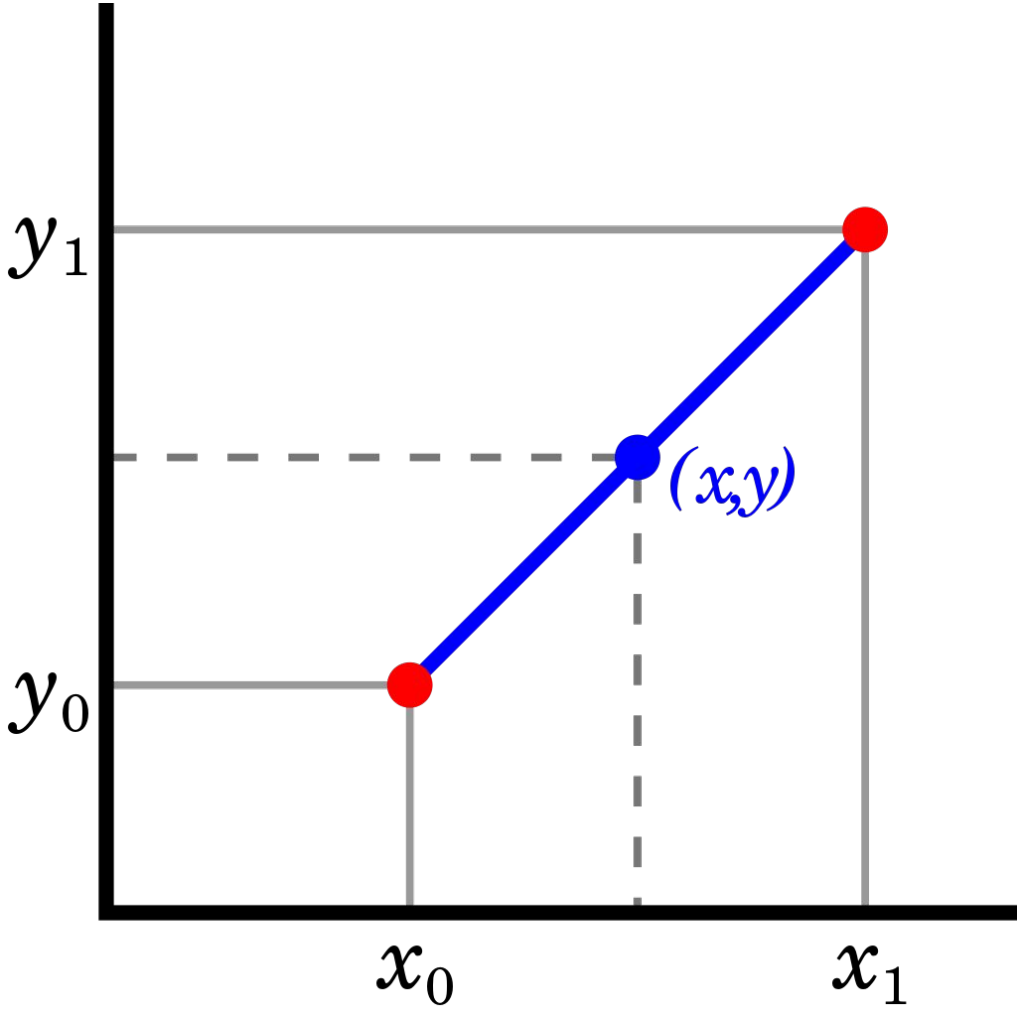
Interpolering

Interpolation is the process of estimating unknown values that fall between known values.

y representerar värdet för en egenskap som ska animeras.

$\mathbf{x_0}$ är tiden när animationen börjar.
$\mathbf{x_1}$ är tiden när animationen ska vara klar.

Datorn räknar ut vilket värde y ska ha med metoden *interpolering*.



Timing-funktioner och interpolering

CSS erbjuder olika **timing-funktioner** som påverkar hur interpolation sker över tid:

- **linear:** Värdet förändras jämnt över tid, vilket följer den linjära interpoleringsmodellen.
- **ease-in, ease-out, cubic-bezier:** Dessa använder mer avancerade kurvmodeller för att beräkna värdet, vilket skapar acceleration eller inbromsning.

Bézierkurva

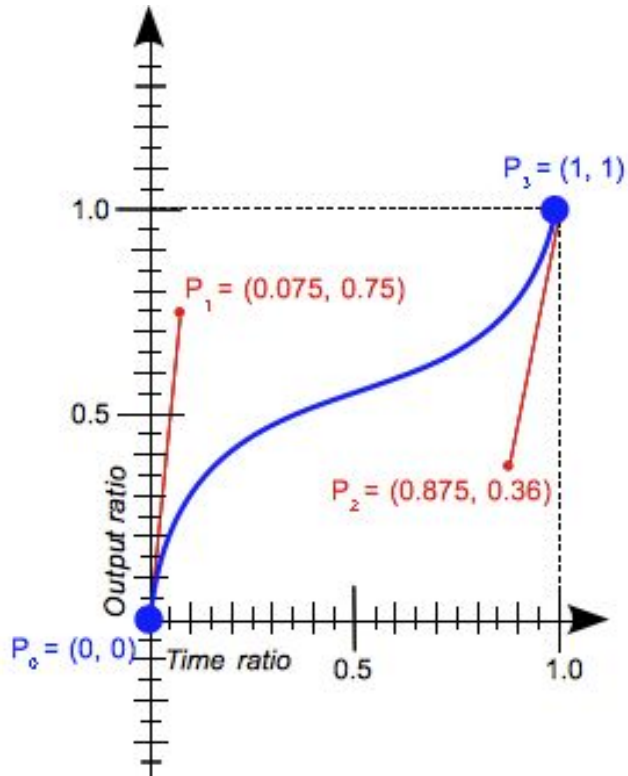
Bézierkurvor beskriver kurvor genom att man sätter kontrollpunkter.

p_0 är alltid (0, 0) och p_3 är alltid (1, 1)

p_1 och p_2 används för att ange två kontrollpunkter som ändrar kurvans form. Exempel:

```
animation-timing-function:
    cubic-bezier(.075,.75,.875,.36);
```

Cubic Bezier ← skapa en kurva online



Några vikiga saker att tänka på när du animerar

- **Prestandaoptimering;** Animationer kan kosta en del särskilt om de påverkar många element samtidigt. Här är några tips för bättre prestanda:
  - hellre “Repaint” än “Reflow”
    - Repaint påverkar endast den visuella renderingens fas (alltså färger, bakgrundsbilder, skuggor etc). Det påverkar inte layoutens struktur.
    - Reflow påverkar layouten och kan orsaka att hela sidan renderas om (alltså typ omplacering av element, förändring av textflöde, eller dimensioner av element).

Fler vikiga saker att tänka på när du animerar

- **Prestandaoptimering;** Animationer kan kosta en del särskilt om de påverkar många element samtidigt. Här är några tips för bättre prestanda:
  - hellre “Repaint” än “Reflow”
    - Repaint påverkar endast den visuella renderingens fas (alltså färger, bakgrundsbilder, skuggor etc). Det påverkar inte layoutens struktur.
    - Reflow påverkar layouten och kan orsaka att hela sidan renderas om (alltså typ omplacering av element, förändring av textflöde, eller dimensioner av element).
- **Undvik överdrivna effekter:** Håll animationer subtila och relevanta för **användarens upplevelse**.
- **Animationer och tillgänglighet;** Dina animationer ska **alltid** respektera “prefers reduced motion”

Tillgänglighet och "Reduced Motion"

När vi använder animationer är det viktigt att tänka på användare som har inställningen **"reduced motion"** aktiverad i sina system. Detta används ofta av personer som kan uppleva yrsel eller illamående av för mycket rörelse.

Viktig regel: Din kod ska alltid lyssna efter inställningen “prefers reduced motion” som automatiskt stänger av eller minskar animationseffekter för användare med den här inställningen.

Exempel:

```css
@media (prefers-reduced-motion: reduce) {
    * {
        animation: none;
        transition: none;
    }
}
```

Lets code!

Animationer med CSS!

Transform

CSS-egenskapen transform omvandlar ett element som har display:block på flera sätt. Vi kan ange flera *transform* efter varandra.

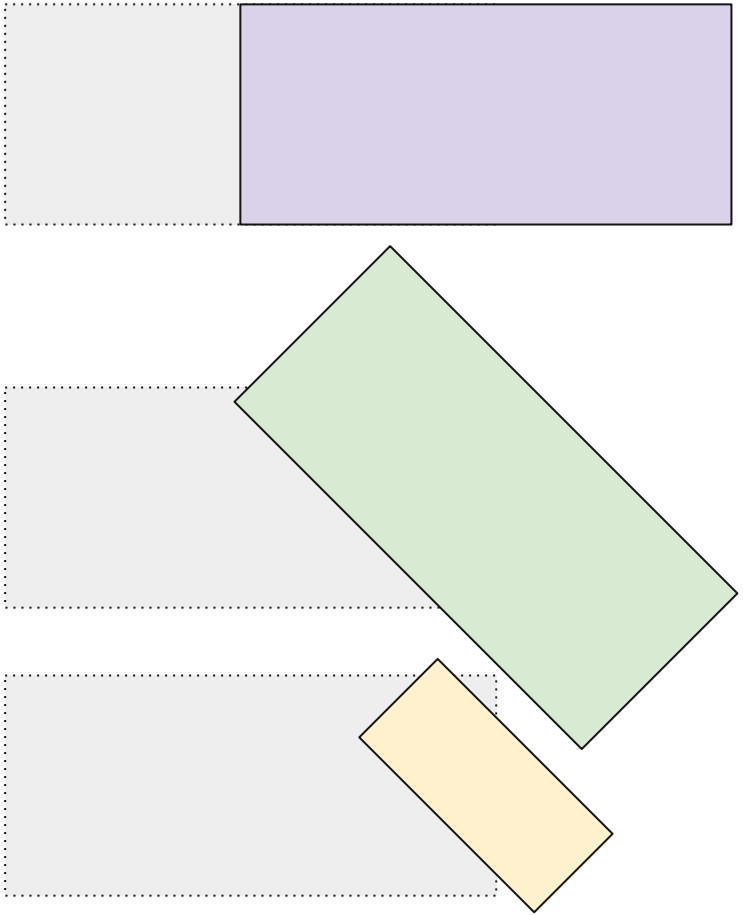
- *translate* (flytta ett antal steg i x- eller y-led)
- *rotate* (rotera ett antal grader medurs)
- *skew* (skevar elementet ett antal grader)
- *scale* (skalning, zooma in/ut)

Exempel transform

```
transform: translate(15px, -50%);
transform: rotate(0.5turn);
transform: skewX(15deg);
transform: scale(1.5, 0.8);
```

Det går att animera transform med transition!

Fler exempel



```
transform: translateX(10em);
```

```
transform: translateX(10em)
    rotate(45deg);
```

```
transform: translateX(10em)
    rotate(45deg) scale(50%);
```

Sammansatta animationer: @keyframes

```
@keyframes moveright {
  0% { left: 0%; }
  70% { left: 80%; }
  100% { left: 70%; }
}
.animated { animation: moveright 2s; }
```

Multi step transitions

```
.box
{ transition:
/* step 1 */
width 1s,
/* step 2 */
background 0.5s 1s; }
```

kommatecken = transitions i flera steg

Fler exempel: Using Multi-Step Animations and Transitions | CSS-Tricks - CSS-Tricks