SVG & Animation

Rcpctition

- nameSpace
- viewBox
- gerenderte Grösse

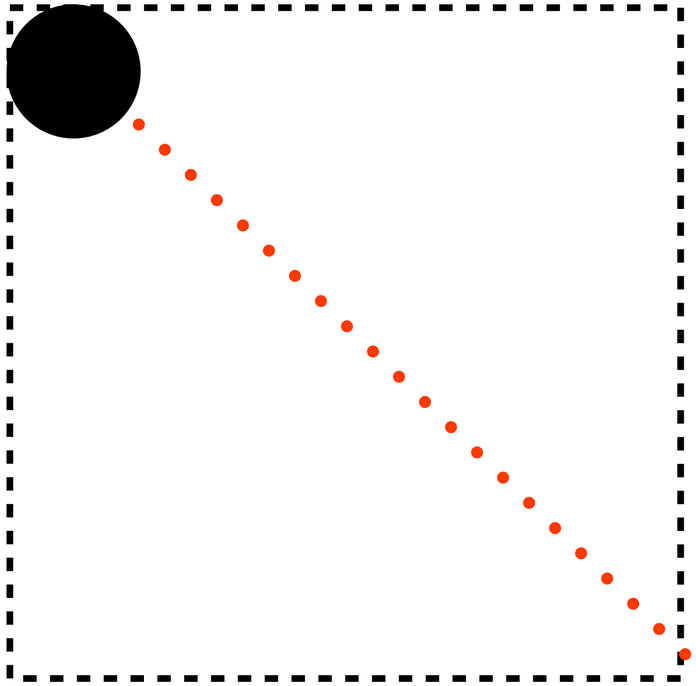
Transformationen

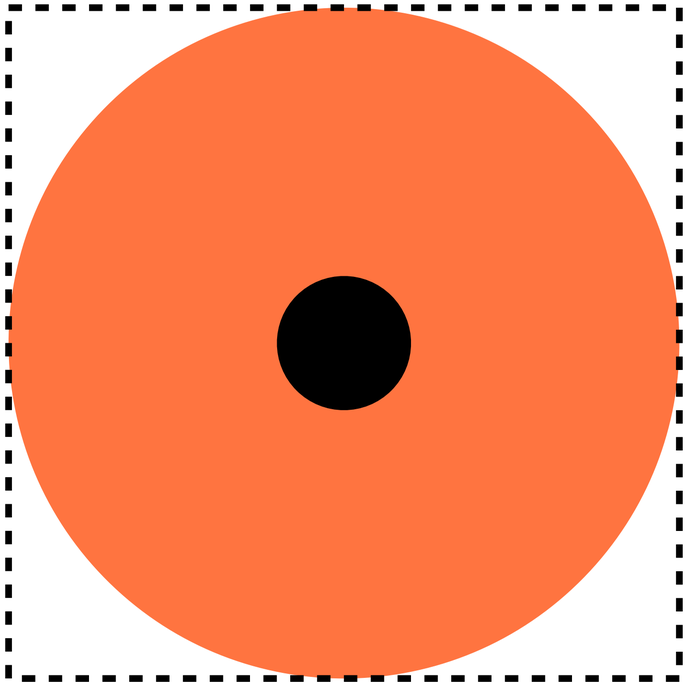
- scale
- translate
- rotate
- (skew)

Unerwartete Nebenwirkungen

- Intuitiv nehmen wir die Skalierung als Veränderung einer Form, eines Objekts wahr.
- Skaliert wird meist das Koordinatensystem.
- Wenn eine Transformation nicht wie erwartet geschieht, könnte das der Grund dafür sein.

Scale

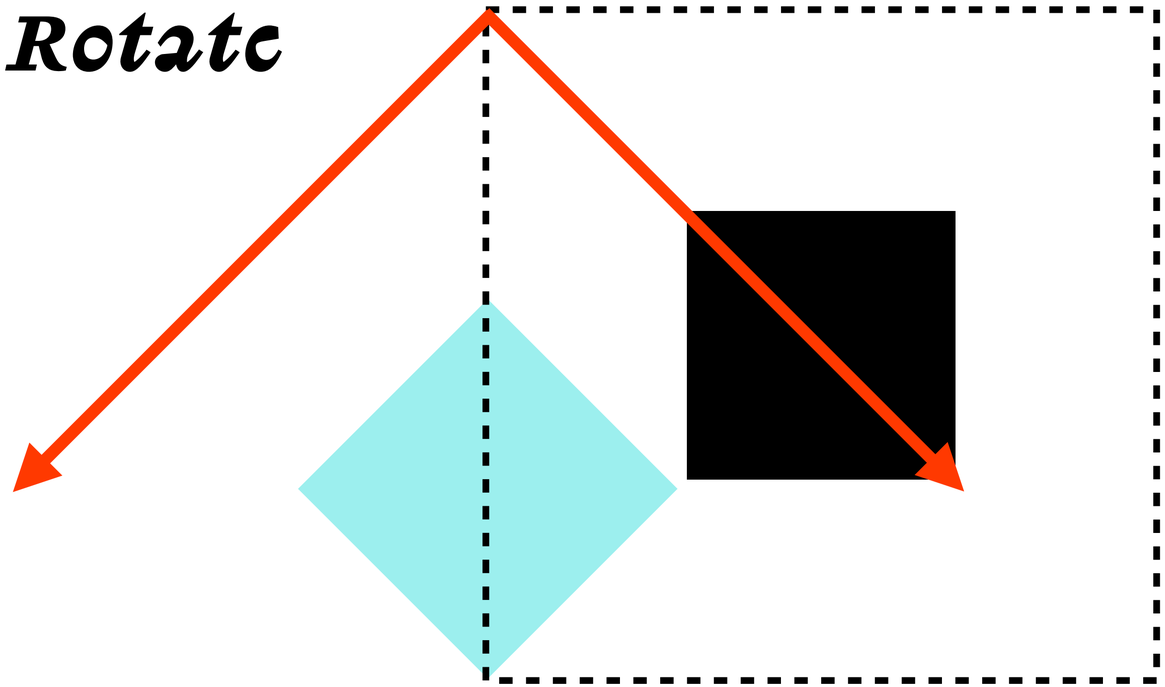




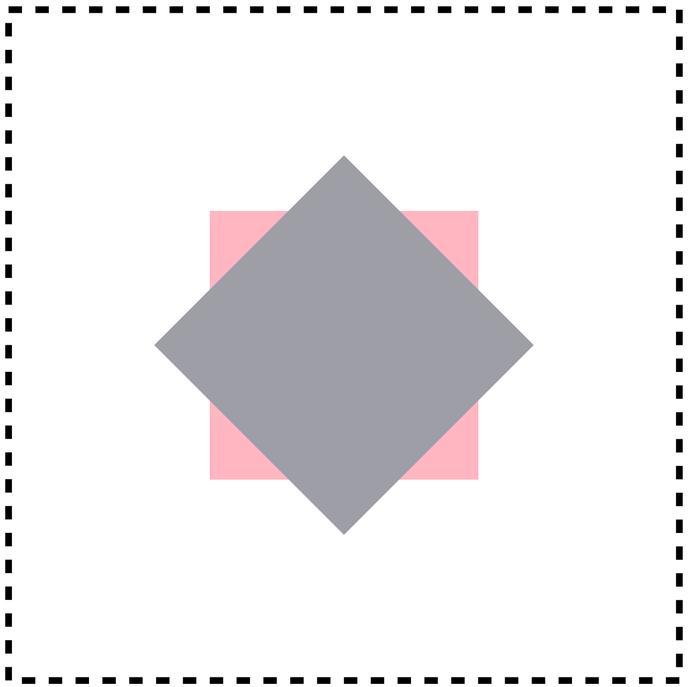
```
transform: scale(5)
transform-origin: 0% 0%
```

```
transform: scale(5)
transform-origin: 50% 50%
```

Rotate

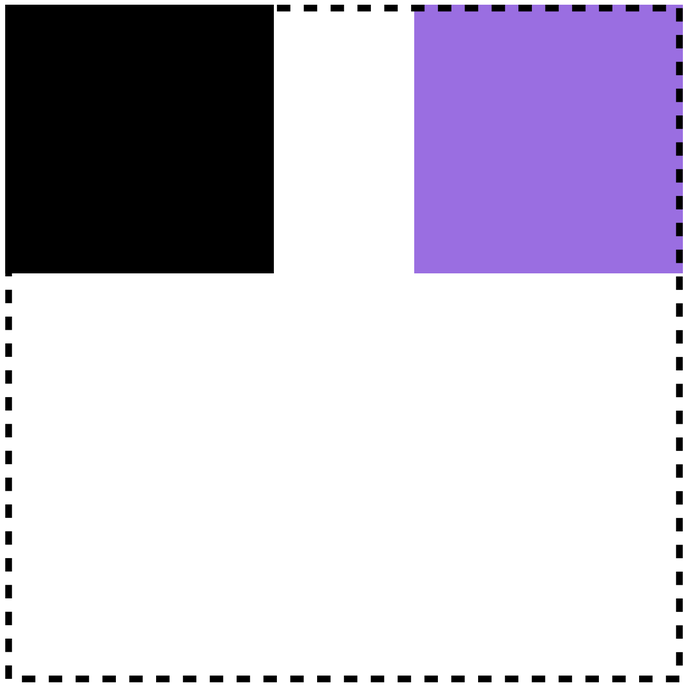


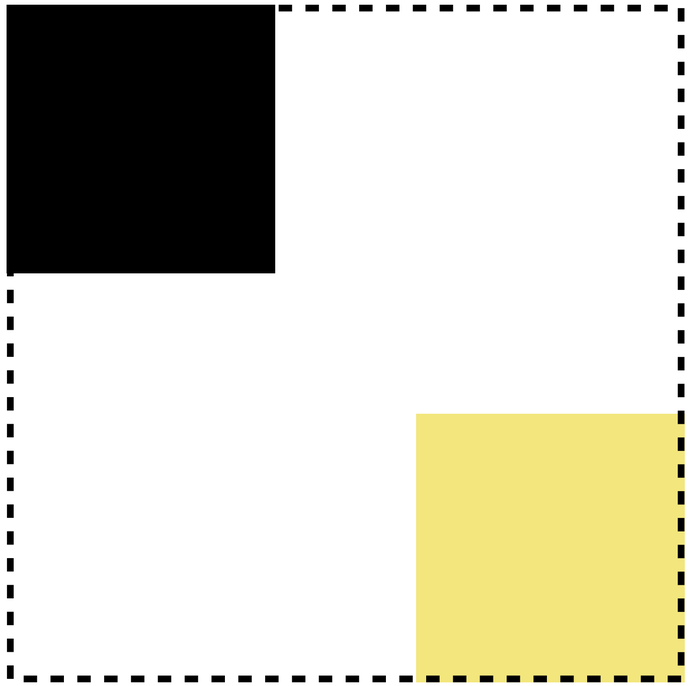
```
transform: rotate(45deg)
transform-origin: 0% 0%
```



```
transform: rotate(45deg)
transform-origin: 50% 50%
```

Translate





`transform: translateX(60%)`

`transform: translate(60%, 60%)`

Die Frage nach dem Ursprung ist hier unerheblich!

Warum Transformation animieren?

- Interpolation einer Transformationen wird direkt durch die Hardware (Grafikkarte) ausgerechnet.
- Das ist viel effizienter, als wenn es von der Software (z.B. Browser) gemacht werden muss.
- Das Gleiche gilt für Deckkraft (opacity).

Verschiedene Arten von Animation

- CSS Animation
- CSS Transition
- JavaScript
- SMIL (Teil von SVG, <animate>)

CSS Animation

- Interpolation wird mit Keyframes definiert.
- Animation meist nicht interaktiv.

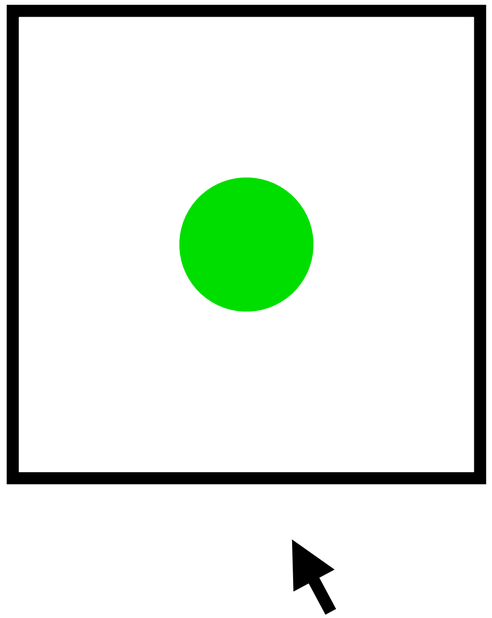
CSS

```css
circle {
  animation-name: hello-bla; /* Name der Keyframes */
  animation-duration: 1000ms; /* Dauer */
  animation-direction: alternate; /* vor und zürück */
  animation-iteration-count: infinite; /* Anzahl Wiederholungen */
}

@keyframes hello-bla {
  from { transform: translateX(0); }
  to { transform: translateX(80%); }
}
```

CSS Transition

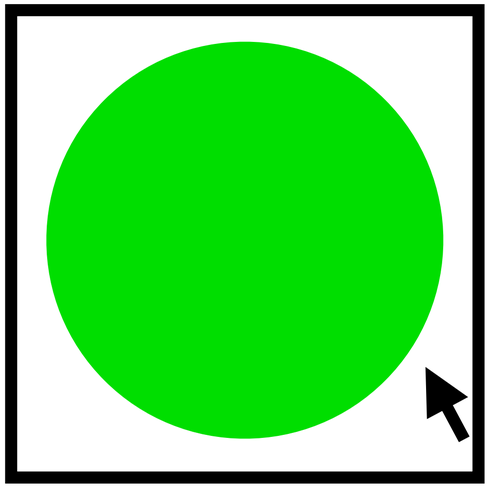
- Interpolation zwischen 2 Zuständen
- Häufig an Interaktion gekoppelt, z.B. Cursor über Element (hover).



CSS

```
circle {
  transition-duration: 300ms;
  transition-property: transform;
  transform-origin: 50% 50%;
}

svg:hover circle {
  transform: scale(5);
}
```



Transition + JavaScript

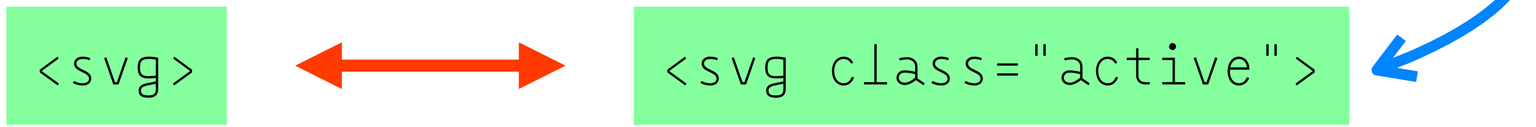
- Im abgegebenen Beispiel aktiviert JavaScript im Sekundentakt eine Klasse.
- Dadurch gelten Regeln im Stylesheet (oder nicht).
- Durch CSS-Transition erfolgt eine Interpolation.

Transition + JS

CSS

```
svg {
  transition-duration: 1000ms;
  transition-property: transform;
}

.active { transform: translateX(50%); }
```



JS

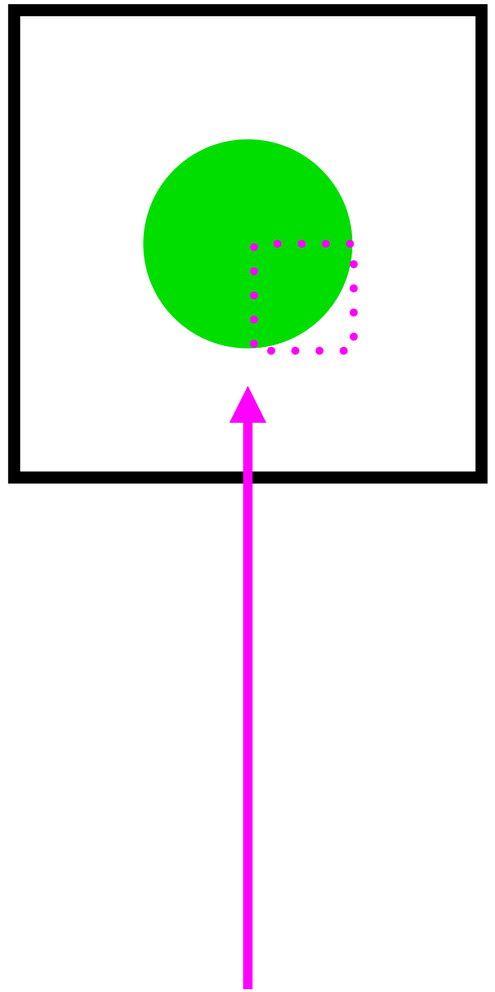
```
setInterval( () => {
  svg.classList.toggle('active')
}, 1000)
```

SMIL

- Animation wird im SVG-Tag <animate> definiert.
- Dadurch ist die Animation Bestandteil von SVG.
- Etwas mühsam: kein ‘transform-origin’
- Link auf MDN

SVG

```
<svg>
    <g transform="translate(500, 500)">
      <circle id="circle-1" cx="0" cy="0" r="100">
        <animateTransform
          attributeName="transform"
          type="scale"
          values="1; 5; 1"
          dur="2s"
          repeatCount="indefinite" />
      </circle>
    </g>
</svg>
```



<g> muss in die Mitte verschoben werden, als Alternative zu fehlendem transform-origin.