Transitions and animations

chapter 6 / 8

Advanced CSS3

Sections in this chapter:

1. Transitions
2. Animations

6-1. Transitions

Normally, if a **style property of an element changes value**, the **change is instantaneous**. *6-1-1*

For example, if we apply this style... *6-1-2*

```
#instantexample:hover {
  transform: rotate(45deg);
}
```

...to this `div`, and then **try hovering over it**...

```
<div id="instantexample"></div>
```

This div is exactly the same as before, but with **one additional rule**: *6-1-3*

```
#transitionexample:hover {
  transition: transform 1s ease;
}
```

```
<div id="transitionexample"></div>
```

Add the code and then try **hovering again**!

What we're really doing is basically saying "Here's two states. **Transition** from the first to the second one in this time". *6-1-4*

As you probably figure, transitions are **a really powerful way to do CSS-only animations**, and **well worth getting to know well**. *6-1-5*

Some notes; *6-1-6*

- The **full shorthand syntax** is `transition: <property> <duration> <timing function> <delay>`

- We can transition **more than one property** by separating them with `,`: `transition: width 2s, height 2s` *6-1-7*

- As with many new / experimental technologies some browsers might require **vendor prefixes** still: `-webkit-transition`, `-ms-transition`, etc. Although the need is growing less and less frequent. *6-1-8*

As per usual there are **lots more detail on MDN**: *6-1-9*

MDN Transition

The **`transform` property** that we used as a transition example also allows elements to be transformed in either 2D or 3D space. *6-1-10*

By the way, the **`transform` property** that we used above is really powerful, and worthy of looking up for its own sake! *6-1-11*

MDN Transform

6-2. Animations

With **transitions** we can do **simple animations from one state to another**. *6-2-1*

But CSS also provides **actual animations** where we **change between predefined frames**!

Behold **this beautiful example**! *6-2-2*

```
#animationexample > div {
  animation-name: looparound;
  animation-duration: 3s;
  animation-iteration-count: infinite;
}
```

```
<div id="animationexample">
  <div></div>
</div>
```

An animation consists of **two parts**. First we **name the animation** and **define the keyframes**, which contain **property values** to be reached at **various points during the animation timeline**. These values will **transition** from one point to the next. *6-2-3*

The earlier animation was **defined like this**: *6-2-4*

```
@keyframes looparound {
  0% {
    top: 0;
    left: 0;
  }
  25% {
    top: 0;
    left: 470px;
  }
  50% {
    top: 170px;
    left: 470px;
  }
  75% {
    top: 170px;
    left: 0;
  }
  100% {
    top: 0;
    left: 0;
  }
}
```

And our element then **used the animation** by specifying **name**, **duration** and **repeats**: *6-2-5*

```
#animationexample > div {
  animation-name: looparound;
  animation-duration: 3s;
  animation-iteration-count: infinite;
}
```

If we only want to **go from one set of property values to another**, then we **might as well just use transitions**. *6-2-6*

But if there are **more than two frames involved** we must **use animations instead**!