CSS Transitions

CSS transitions allows you to change property values smoothly, over a given duration.

Mouse over the element below to see a CSS transition effect:

CSS

In this chapter you will learn about the following properties:

- transition
- transition-delay
- transition-duration
- transition-property
- transition-timing-function

How to Use CSS Transitions?

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

Note: If the duration part is not specified, the transition will have no effect, because the default value is 0.

The following example shows a 100px * 100px red element. The element has also specified a transition effect for the width property, with a duration of 2 seconds:

Example

```
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}
```

The transition effect will start when the specified CSS property (width) changes value.

Now, let us specify a new value for the width property when a user mouses over the element:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background: red;
 transition: width 2s;
}

div:hover {
 width: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>

<p>Hover over the div element below, to see the transition effect:</p>
<div></div>

</body>
</html>
```

Notice that when the cursor mouses out of the element, it will gradually change back to its original style.

Change Several Property Values

The following example adds a transition effect for both the

width and height property, with a duration of 2 seconds for the width and 4 seconds for the height:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background: red;
 transition: width 2s, height 4s;
}

div:hover {
 width: 300px;
 height: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>

<p>Hover over the div element below, to see the transition effect:</p>

<div></div>

</body>
</html>
```

Specify the Speed Curve of the Transition

The transition-timing-function property specifies the speed

curve of the transition effect.

The transition-timing-function property can have the following values:

- ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
- linear - specifies a transition effect with the same speed from start to end
- ease-in - specifies a transition effect with a slow start
- ease-out - specifies a transition effect with a slow end
- ease-in-out - specifies a transition effect with a slow start and end
- cubic-bezier(n,n,n,n) - lets you define your own values in a cubic-bezier function

The following example shows some of the different speed curves that can be used:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background: red;
 transition: width 2s;
}

#div1 {transition-timing-function: linear;}
#div2 {transition-timing-function: ease;}
#div3 {transition-timing-function: ease-in;}
#div4 {transition-timing-function: ease-out;}
#div5 {transition-timing-function: ease-in-out;}

div:hover {
 width: 300px;
```

```
}
</style>
</head>
<body>

<h1>The transition-timing-function Property</h1>

<p>Hover over the div elements below, to see the different
speed curves:</p>

<div id="div1">linear</div><br>
<div id="div2">ease</div><br>
<div id="div3">ease-in</div><br>
<div id="div4">ease-out</div><br>
<div id="div5">ease-in-out</div><br>

</body>
</html>
```

Delay the Transition Effect

The transition-delay property specifies a delay (in seconds) for the transition effect.

The following example has a 1 second delay before starting:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background: red;
```

```
 transition: width 3s;
 transition-delay: 1s;
}

div:hover {
 width: 300px;
}
</style>
</head>
<body>

<h1>The transition-delay Property</h1>

<p>Hover over the div element below, to see the transition effect:</p>

<div></div>

<p><b>Note:</b> The transition effect has a 1 second delay before starting.</p>

</body>
</html>
```

Transition + Transformation

The following example adds a transition effect to the transformation:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
```

```
 background: red;
 transition: width 2s, height 2s, transform 2s;
}

div:hover {
 width: 300px;
 height: 300px;
 transform: rotate(180deg);
}
</style>
</head>
<body>

<h1>Transition + Transform</h1>

<p>Hover over the div element below:</p>

<div></div>

</body>
</html>
```

More Transition Examples

The CSS transition properties can be specified one by one, like this:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background: red;
 transition-property: width;
```

```
 transition-duration: 2s;
 transition-timing-function: linear;
 transition-delay: 1s;
}

div:hover {
 width: 300px;
}
</style>
</head>
<body>

<h1>The transition Properties Specified One by One</h1>

<p>Hover over the div element below, to see the transition
effect:</p>

<div></div>

<p><b>Note:</b> The transition effect has a 1 second delay
before starting.</p>

</body>
</html>
```

CSS Animations

CSS allows animation of HTML elements without using JavaScript or Flash!

CSS

In this chapter you will learn about the following properties:

- @keyframes
- animation-name
- animation-duration

- animation-delay
- animation-iteration-count
- animation-direction
- animation-timing-function
- animation-fill-mode
- animation
- What are CSS Animations?
- An animation lets an element gradually change from one style to another.
- You can change as many CSS properties you want, as many times as you want.
- To use CSS animation, you must first specify some keyframes for the animation.
- Keyframes hold what styles the element will have at certain times.
-

- The @keyframes Rule
- When you specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at certain times.
- To get an animation to work, you must bind the animation to an element.
- The following example binds the "example" animation to the element. The animation will last for 4 seconds, and it will gradually change the background-color of the element from "red" to "yellow":

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
```

```
 background-color: red;
 animation-name: example;
 animation-duration: 4s;
}

@keyframes example {
 from {background-color: red;}
 to {background-color: yellow;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes
back to its original style.</p>

</body>
</html>
```

Note: The animation-duration property defines how long an animation should take to complete. If the animation-duration property is not specified, no animation will occur, because the default value is 0s (0 seconds).

In the example above we have specified when the style will change by using the keywords "from" and "to" (which represents 0% (start) and 100% (complete)).

It is also possible to use percent. By using percent, you can add as many style changes as you like.

The following example will change the background-color of the element when the animation is 25% complete, 50% complete, and again when the animation is 100% complete:

```
<!DOCTYPE html>
<html>
<head>
```

```
<style>
div {
 width: 100px;
 height: 100px;
 background-color: red;
 animation-name: example;
 animation-duration: 4s;
}

@keyframes example {
 0%   {background-color: red;}
 25%  {background-color: yellow;}
 50%  {background-color: blue;}
 100% {background-color: green;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes
back to its original style.</p>

</body>
</html>
```

The following example will change both the background-color and the position of the element when the animation is 25% complete, 50% complete, and again when the animation is 100% complete:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background-color: red;
 position: relative;
 animation-name: example;
 animation-duration: 4s;
}

@keyframes example {
 0%   {background-color:red; left:0px; top:0px;}
 25%  {background-color:yellow; left:200px; top:0px;}
 50%  {background-color:blue; left:200px; top:200px;}
 75%  {background-color:green; left:0px; top:200px;}
 100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes
back to its original style.</p>

</body>
```

```
</html>
```

Delay an Animation

The animation-delay property specifies a delay for the start of an animation.

The following example has a 2 seconds delay before starting the animation:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background-color: red;
 position: relative;
 animation-name: example;
 animation-duration: 4s;
 animation-delay: 2s;
}

@keyframes example {
 0%   {background-color:red; left:0px; top:0px;}
 25%  {background-color:yellow; left:200px; top:0px;}
 50%  {background-color:blue; left:200px; top:200px;}
 75%  {background-color:green; left:0px; top:200px;}
 100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>
```

The animation-delay property specifies a delay for the start of an animation. The following example has a 2 seconds delay before starting the animation:

</body>
</html>

Negative values are also allowed. If using negative values, the animation will start as if it had already been playing for *N* seconds.

In the following example, the animation will start as if it had already been playing for 2 seconds:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background-color: red;
 position: relative;
 animation-name: example;
 animation-duration: 4s;
 animation-delay: -2s;
}

@keyframes example {
 0%   {background-color:red; left:0px; top:0px;}
 25%  {background-color:yellow; left:200px; top:0px;}
 50%  {background-color:blue; left:200px; top:200px;}
 75%  {background-color:green; left:0px; top:200px;}
 100% {background-color:red; left:0px; top:0px;}
```

```
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>Using negative values in the animation-delay property:
Here, the animation will start as if it had already been playing
for 2 seconds:</p>

<div></div>

</body>
</html>
```

Set How Many Times an Animation Should Run

The animation-iteration-count property specifies the number of times an animation should run.

The following example will run the animation 3 times before it stops:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background-color: red;
 position: relative;
 animation-name: example;
 animation-duration: 4s;
 animation-iteration-count: 3;
}
```

```
@keyframes example {
 0%   {background-color:red; left:0px; top:0px;}
 25%  {background-color:yellow; left:200px; top:0px;}
 50%  {background-color:blue; left:200px; top:200px;}
 75%  {background-color:green; left:0px; top:200px;}
 100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>The animation-iteration-count property specifies the
number of times an animation should run. The following
example will run the animation 3 times before it stops:</p>

<div></div>

</body>
</html>
```

Run Animation in Reverse Direction or Alternate Cycles

The animation-direction property specifies whether an animation should be played forwards, backwards or in alternate cycles.

The animation-direction property can have the following values:

- normal - The animation is played as normal (forwards). This is default
- reverse - The animation is played in reverse direction (backwards)

- alternate - The animation is played forwards first, then backwards
- alternate-reverse - The animation is played backwards first, then forwards

The following example will run the animation in reverse direction (backwards):

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
 height: 100px;
 background-color: red;
 position: relative;
 animation-name: example;
 animation-duration: 4s;
 animation-direction: reverse;
}

@keyframes example {
 0%   {background-color:red; left:0px; top:0px;}
 25%  {background-color:yellow; left:200px; top:0px;}
 50%  {background-color:blue; left:200px; top:200px;}
 75%  {background-color:green; left:0px; top:200px;}
 100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>
```

The animation-direction property specifies whether an animation should be played forwards, backwards or in alternate cycles. The following example will run the animation in reverse direction (backwards):

</body>

</html>

Specify the Speed Curve of the Animation

The animation-timing-function property specifies the speed curve of the animation.

The animation-timing-function property can have the following values:

- ease - Specifies an animation with a slow start, then fast, then end slowly (this is default)
- linear - Specifies an animation with the same speed from start to end
- ease-in - Specifies an animation with a slow start
- ease-out - Specifies an animation with a slow end
- ease-in-out - Specifies an animation with a slow start and end
- cubic-bezier(n,n,n,n) - Lets you define your own values in a cubic-bezier function

The following example shows some of the different speed curves that can be used:

<!DOCTYPE html>

<html>

<head>

<style>

```
div {
 width: 100px;
 height: 50px;
 background-color: red;
 font-weight: bold;
 position: relative;
 animation: mymove 5s infinite;
}

#div1 {animation-timing-function: linear;}
#div2 {animation-timing-function: ease;}
#div3 {animation-timing-function: ease-in;}
#div4 {animation-timing-function: ease-out;}
#div5 {animation-timing-function: ease-in-out;}

@keyframes mymove {
 from {left: 0px;}
 to {left: 300px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>The animation-timing-function property specifies the
speed curve of the animation. The following example shows
some of the different speed curves that can be used:</p>

<div id="div1">linear</div>
<div id="div2">ease</div>
<div id="div3">ease-in</div>
<div id="div4">ease-out</div>
```

```
<div id="div5">ease-in-out</div>


</body>
</html>
```

Specify the fill-mode For an Animation

CSS animations do not affect an element before the first keyframe is played or after the last keyframe is played. The animation-fill-mode property can override this behavior.

The animation-fill-mode property specifies a style for the target element when the animation is not playing (before it starts, after it ends, or both).

The animation-fill-mode property can have the following values:

- none - Default value. Animation will not apply any styles to the element before or after it is executing
- forwards - The element will retain the style values that is set by the last keyframe (depends on animation-direction and animation-iteration-count)
- backwards - The element will get the style values that is set by the first keyframe (depends on animation-direction), and retain this during the animation-delay period
- both - The animation will follow the rules for both forwards and backwards, extending the animation properties in both directions

The following example lets the element retain the style values from the last keyframe when the animation ends:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
```

```
 height: 100px;
 background: red;
 position: relative;
 animation-name: example;
 animation-duration: 3s;
 animation-fill-mode: forwards;
}

@keyframes example {
 from {top: 0px;}
 to {top: 200px; background-color: blue;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>Let the div element retain the style values set by the last
keyframe when the animation ends:</p>

<div></div>

</body>
</html>



<!DOCTYPE html>
<html>
<head>
<style>
div {
 width: 100px;
```

```
 height: 100px;
 background: red;
 position: relative;
 animation-name: example;
 animation-duration: 3s;
 animation-delay: 2s;
 animation-fill-mode: both;
}

@keyframes example {
 from {top: 0px; background-color: yellow;}
 to {top: 200px; background-color: blue;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>Let the div element get the style values set by the first
keyframe before the animation starts, and retain the style
values from the last keyframe when the animation ends:</p>

<div></div>

</body>
</html>
```