**animation**

The animation property in CSS can be used to animate many other CSS properties such as [color](https://css-tricks.com/almanac/properties/c/color/), [background-color](https://css-tricks.com/almanac/properties/b/background-color/), [height](https://css-tricks.com/almanac/properties/h/height/), or [width](https://css-tricks.com/almanac/properties/w/width/). Each animation needs to be defined with the @keyframes [at-rule](https://css-tricks.com/at-rule-css/) which is then called with the animation property, like so:

Each @keyframes at-rule defines what should happen at specific moments during the animation. For example, **0% is the beginning of the animation** and **100% is the end**. These keyframes can then be controlled either by the shorthand animation property, or its **eight sub-properties**, to give more control over how those keyframes should be manipulated.

[**#**](https://css-tricks.com/almanac/properties/a/animation/#article-header-id-0)**Sub-properties**

* animation-name: declares the name of the @keyframes at-rule to manipulate.
* animation-duration: the length of time it takes for an animation to complete one cycle.
* animation-timing-function: establishes preset acceleration curves such as ease or linear.
* animation-delay: the time between the element being loaded and the start of the animation sequence.
* animation-direction: sets the direction of the animation after the cycle. Its default resets on each cycle.
* animation-iteration-count: the number of times the animation should be performed.
* animation-fill-mode: sets which values are applied before/after the animation.  
  For example, you can set the last state of the animation to remain on screen, or you can set it to switch back to before when the animation began.
* animation-play-state: pause/play the animation.

These sub-properties can then be used like so:

@keyframes stretch {

/\* declare animation actions here \*/

}

.element {

animation-name: stretch;

animation-duration: 1.5s;

animation-timing-function: ease-out;

animation-delay: 0s;

animation-direction: alternate;

animation-iteration-count: infinite;

animation-fill-mode: none;

animation-play-state: running;

}

/\*

is the same as:

\*/

.element {

animation:

stretch

1.5s

ease-out

0s

alternate

infinite

none

running;

}

|  |  |
| --- | --- |
| animation-timing-function | ease, ease-out, ease-in, ease-in-out, linear, cubic-bezier(x1, y1, x2, y2) (e.g. cubic-bezier(0.5, 0.2, 0.3, 1.0)) |
| animation-duration | Xs or Xms |
| animation-delay | Xs or Xms |
| animation-iteration-count | X |
| animation-fill-mode | forwards, backwards, both, none |
| animation-direction | normal, alternate, reverse |
| animation-play-state | paused, running |

### Multiple steps

If an animation has the same starting and ending properties, it's useful to comma-separate the 0% and 100% values inside @keyframes:

@keyframes pulse {

0%, 100% {

background-color: yellow;

}

50% {

background-color: red;

}

}

### Multiple animations

You can comma-separate the values to declare multiple animations on a selector as well. In the example below, we want to change the color of the circle in a @keyframe whilst also nudging it from side to side with another.

.element {

animation:

pulse 3s ease infinite alternate,

nudge 5s linear infinite alternate;

}