@keyframes Правило @keyframes устанавливает ключевые кадры при анимации элемента. Ключевой кадр это свойства элемента (прозрачность, цвет, положение и др.), которые должны применяться к элементу в заданный момент времени. Таким образом, анимация представляет собой плавный переход стилевых свойств от одного ключевого кадра к другому. Вычисление промежуточных значений между такими кадрами берёт на себя браузер.

***@keyframes box { from { left: 0; } to { left: 300px; } }  
@keyframes box { 50% { left: 0; } 90% { left: 300px; } }***

Animation Универсальное свойство, которое задаёт сразу несколько параметров анимации.

Значение по умолчанию

1. animation-name: none
2. animation-duration: 0s
3. animation-timing-function: ease
4. animation-delay: 0s
5. animation-iteration-count: 1
6. animation-direction: normal
7. animation-fill-mode: none
8. animation-play-state: running

*animation: animation-duration || animation-timing-function || animation-delay || animation-iteration-count || animation-direction || animation-fill-mode || animation-play-state || [none | animation-name]*

***.creature::before, .creature::after {  
animation: eye 3s ease-in-out infinite;  
 -webkit-animation: eye 3s ease-in-out infinite;}***

animation-duration Задаёт время в секундах или миллисекундах, сколько должен длиться один цикл анимации. По умолчанию значение равно 0s, это означает, что никакой анимации нет.

Можно указать несколько значений, перечисляя их через запятую. Отрицательные значения и значения без указания единиц времени (s или ms) недопустимы и будут игнорироваться.

*animation-duration: <время> [,<время>]\*  
animation-duration: 1s;*

***.fadeInLeft { -webkit-animation-name: fadeInLeft; -webkit-animation-duration: 1s;  
animation-name: fadeInLeft; animation-duration: 1s; }***

animation-timing-function Устанавливает, согласно какой функции времени должна происходить анимация каждого цикла между ключевыми кадрами. Она представляет собой математическую функцию, показывающую, как быстро по времени меняется значение свойства. Начальная точка имеет координаты 0.0, 0.0, конечная — 1.0, 1.0, при этом функция по оси ординат может превышать эти значения в большую или меньшую сторону (рис. 1).

*animation-timing-function: ease | ease-in | ease-out | ease-in-out | linear | step-start | step-end | steps | cubic-bezier*

1. ease Анимация начинается медленно, затем ускоряется и к концу движения опять замедляется. Аналогично cubic-bezier(0.25,0.1,0.25,1).
2. ease-in Анимация медленно начинается, к концу ускоряется. Аналогично cubic-bezier(0.42,0,1,1).
3. ease-out Анимация начинается быстро, к концу замедляется. Аналогично cubic-bezier(0,0,0.58,1).
4. ease-in-out Анимация начинается и заканчивается медленно. Аналогично cubic-bezier(0.42,0,0.58,1).
5. linear Одинаковая скорость от начала и до конца.
6. step-start Как таковой анимации нет. Стилевые свойства сразу же принимают конечное значение.
7. step-end Как таковой анимации нет. Стилевые свойства находятся в начальном значении заданное время, затем сразу же принимают конечное значение.
8. steps Ступенчатая функция, имеющая заданное число шагов: *animation-timing-function: steps(<число>, start | end)*
9. Здесь: <число> — целое число больше нуля;
10. start — задаёт полунепрерывную снизу функцию;
11. end — задаёт полунепрерывную сверху функцию.
12. cubic-bezier Задаёт функцию движения в виде кривой Безье.

***.progress::before{ animation: progress 5s; animation-timing-function: linear;  
 animation-duration: 5s; animation-fill-mode: forwards;  
-webkit-animation-name: progress; -webkit-animation-timing-function: linear;}***

animation-delay Свойство animation-delay устанавливает время ожидания перед запуском цикла анимации. Значение 0s или 0ms запускает анимацию сразу же. Отрицательное значение также включает анимацию без задержек, но может привести к изменению вида начала анимации.

Допустимо указывать несколько значений, перечисляя их через запятую.

*animation-delay: <время> [,<время>]\**

***.fadeInTop { -webkit-animation-name: fadeInTop;  
 -webkit-animation-duration: 1s; -webkit-animation-delay: 2s;}***

animation-iteration-count Свойство определяет, сколько раз проигрывать цикл анимации до её остановки.

1. infinite Анимация повторяется бесконечно.
2. <число> Сколько раз должна повторяться анимация. Отрицательные значения не допустимы. Можно использовать числа меньше единицы, для примера 0.5 будет означать половину цикла анимации.

***.blink { -webkit-animation: blink 1s; -webkit-animation-iteration-count: infinite;}***

animation-direction Устанавливает направление движения анимации.

*animation-direction: normal | alternate | reverse | alternate-reverse*

1. normal Анимация идёт с самого начала, после завершения цикла возвращается к исходному состоянию.
2. alternate Анимация идёт с начала до конца, затем плавно возвращается в исходное положение.
3. reverse Анимация идёт с конца цикла, после его завершения возвращается к исходному состоянию.
4. alternate-reverse Анимация идёт с конца до начала, затем плавно возвращается в исходное положение.

*.alternate { top: 100px; animation-direction: alternate; background-color: #fee7dc; }*

animation-fill-mode Определяет, какие стили должны применяться к элементу, когда анимация не проигрывается. Например, после её завершения или при остановке. По умолчанию, в момент окончания анимации стиль элемента возвращается к исходному, свойство animation-fill-mode позволяет изменить это поведение и сделать так, чтобы стиль элемента оставался как у последнего ключевого кадра.

К примеру, если вы делаете выезжающее из-за края окна браузера сообщение, то после окончания анимации сообщение вернётся обратно за край экрана. Значение forwards свойства animation-fill-mode изменяет это поведение и оставляет стили на момент завершения движения. Таким образом, сообщение плавно выдвинется из-за края окна и останется на месте.

*animation-fill-mode: none | forwards | backwards | both*

1. none К элементу не применяются какие-либо стили.
2. forwards К элементу по окончанию анимации применяется стиль последнего ключевого кадра. Каким будет этот кадр в последовательности анимации зависит от комбинации значений свойств animation-direction и animation-iteration-count (табл. 1).
3. backwards К элементу применяется стиль первого ключевого кадра и он остаётся на протяжении периода заданного animation-delay. Первый ключевой кадр определяется на основании значения animation-direction (табл. 2).
4. both К элементу применяются оба правила, как для forwards, так и для backwards.

Таблица 1

animation-direction animation-iteration-count Значение

1. normal любое число 100% или to
2. reverse любое число 0% или from
3. alternate чётное число 0% или from
4. alternate нечётное число 100% или to
5. alternate-reverse чётное число 100% или to
6. alternate-reverse нечётное число 0% или from

Таблица 2

animation-direction Значение

1. normal или alternate 0% или from
2. reverse или alternate-reverse 100% или to

***.wheel { animation: wheel 3s 1s; animation-fill-mode: both;  
-webkit-animation: wheel 3s 1s; -webkit-animation-fill-mode: both; }***

animation-play-state Свойство определяет, проигрывать анимацию или поставить её на паузу. При продолжении установленной на паузе анимации она начинается с того момента где была остановлена.

*animation-play-state: running | paused*

1. running Проигрывать анимацию.
2. paused Поставить анимацию на паузу.

*.marquee:hover { -webkit-animation-play-state: paused; animation-play-state: paused; /\* Пауза \*/ }*

animation-name Устанавливает одну или несколько анимаций, которые применяются к элементу. Представляет собой имя, связанное с правилом @keyframes, оно в свою очередь задаёт последовательность движения.

*animation-name: none | <идентификатор> [, none | <идентификатор>]\**

1. none Отключает анимацию.
2. <идентификатор> Текстовая строка, которая связана с анимацией. Идентификатор должен начинаться с латинской буквы или подчёркивания (\_), также может содержать цифры от 0 до 9 и дефис (-). Запрещено использовать зарезервированные ключевые слова none, unset, inherit, initial.

***.an {-webkit-animation-name: spaceOutUp;  
 animation-name: spaceOutUp;  
 -webkit-animation-duration: 1s;  
 animation-duration: 1s; }***

transition Универсальное свойство, которое позволяет одновременно задать значения transition-property, transition-duration, transition-timing-function и transition-delay. Устанавливает эффект перехода между двумя состояниями элемента, они могут быть определены с помощью псевдокласса :hover или :active, а также динамически через JavaScript.

*transition: <переход> [, <переход> ]\**

*<переход> = [ none | <transition-property> ] || <transition-duration> || <transition-timing-function> || <transition-delay>*

1. none Отменяет эффект перехода.

***#bar { top:-5.5em; right:5em; /\* Положение \*/ transition: top 1s ease-out 0.5s;/\* Переход \*/ }***

transition-delay устанавливает время ожидания перед запуском эффекта перехода. Значение 0sзапускает анимацию сразу же. Отрицательное значение также включает анимацию без задержек, но может привести к изменению вида начала анимации.

Допустимо указывать несколько значений, перечисляя их через запятую. Каждое значение будет применяться к свойству, заданному в параметрах transition-property.

*transition-delay: <время> [,<время>]\**

***#bar { top:-5.5em; right:5em; /\* Положение \*/  
transition: .4s ease-out; transition-delay: 0.5s; /\* Анимация \*/***

transition-duration Задаёт время в секундах или миллисекундах, сколько должна длиться анимация перехода до её завершения. По умолчанию значение равно 0s, это означает, что никакой анимации нет, переход происходит мгновенно.

Можно указать несколько значений, перечисляя их через запятую. Каждое значение применяется к свойствам, заданным через transition-property.

*transition-duration: <время> [,<время>]\**

*.warn { padding: 20px; transition-duration: 2s;}*

transition-property Устанавливает имя стилевого свойства, значение которого будет отслеживаться для создания эффекта перехода.

*transition-property: none | all | <свойство> [,<свойство>]\**

1. none Никакое свойство не задано.
2. all Все свойства будут отслеживаться.
3. <свойство> Название стилевого свойства, регистр при его написании не учитывается. При указании нескольких свойств они перечисляются друг за другом через запятую.

***.menu { position: absolute; height: 100px; width: 100%; background: #fc0;  
border-bottom: 20px solid #333; top: -100px;  
/\* Анимация \*/ transition-property: top; transition-duration: 2s; }***

transition-timing-function Устанавливает, насколько быстро должно изменяться значение стилевого свойство для которого применяется эффект перехода.

transition-timing-function представляет собой математическую функцию, показывающую, как быстро по времени меняется указанное через transition-property значение свойства. Начальная точка имеет координаты 0.0, 0.0, конечная — 1.0, 1.0, при этом функция по оси ординат может превышать эти значения в большую или меньшую сторону (рис. 1).

*transition-timing-function: ease | ease-in | ease-out | ease-in-out | linear | step-start | step-end | steps | cubic-bezier*

1. ease Анимация начинается медленно, затем ускоряется и к концу движения опять замедляется. Аналогично cubic-bezier(0.25,0.1,0.25,1).
2. ease-in Анимация медленно начинается, к концу ускоряется. Аналогично cubic-bezier(0.42,0,1,1).
3. ease-out Анимация начинается быстро, к концу замедляется. Аналогично cubic-bezier(0,0,0.58,1).
4. ease-in-out Анимация начинается и заканчивается медленно. Аналогично cubic-bezier(0.42,0,0.58,1).
5. linear Одинаковая скорость от начала и до конца.
6. step-start Как таковой анимации нет. Стилевые свойства сразу же принимают конечное значение.
7. step-end Как таковой анимации нет. Стилевые свойства находятся в начальном значении заданное время, затем сразу же принимают конечное значение.
8. steps Ступенчатая функция, имеющая заданное число шагов: transition-timing-function: steps(<число>, start | end) Здесь: <число> — целое число больше нуля; start — задаёт полунепрерывную снизу функцию; end — задаёт полунепрерывную сверху функцию
9. cubic-bezier Задаёт функцию движения в виде кривой Безье

*.progress::before{ transition-timing-function: linear; transition-duration: 5s; content: '';}*

transition-property Устанавливает имя стилевого свойства, значение которого будет отслеживаться для создания эффекта перехода.

*transition-property: none | all | <свойство> [,<свойство>]\**

1. none Никакое свойство не задано.
2. all Все свойства будут отслеживаться.
3. <свойство> Название стилевого свойства, регистр при его написании не учитывается. При указании нескольких свойств они перечисляются друг за другом через запятую.

***.menu { position: absolute; height: 100px; width: 100%;  
background: #fc0; border-bottom: 20px solid #333; top: -100px;  
/\* Анимация \*/ transition-property: top; transition-duration: 2s;}***

Cursor Устанавливает форму курсора, когда он находится в пределах элемента. Вид курсора зависит от операционной системы и установленных параметров.

*cursor: [url('адрес'),] | [ <курсор> ]*

1. url Позволяет установить свой собственный курсор, для этого нужно указать путь к файлу с курсором.
2. auto Вид курсора по умолчанию для текущего элемента.
3. none Отключает отображение курсора.

Остальные допустимые значения приведены в табл. 1. Чтобы увидеть как курсор выглядит в реальности, наведите на колонку Тест.

Табл. 1. Вид курсора

Вид Значение Тест Пример Примечание

default P {cursor: default}   
context-menu P {cursor: context-menu} Отображается только в IE10+.  
help P {cursor: help}   
pointer P {cursor: pointer}  
progress P {cursor: progress}  
wait P {cursor: wait}   
cell P {cursor: cell}   
crosshair P {cursor: crosshair}   
text P {cursor: text}   
vertical-text P {cursor: vertical-text}   
alias P {cursor: alias}   
copy P {cursor: copy}   
move P {cursor: move}   
no-drop P {cursor: no-drop} Все браузеры, кроме IE, отображают как not-allowed.  
not-allowed P {cursor: not-allowed}   
all-scroll P {cursor: all-scroll}   
col-resize P {cursor: col-resize}   
row-resize P {cursor: row-resize}   
n-resize P {cursor: n-resize}   
ne-resize P {cursor: ne-resize}  
e-resize P {cursor: e-resize}   
se-resize P {cursor: se-resize}   
s-resize P {cursor: s-resize}   
sw-resize P {cursor: sw-resize}   
w-resize P {cursor: w-resize}   
nw-resize P {cursor: nw-resize}   
nesw-resize P {cursor: nesw-resize}   
nwse-resize P {cursor: nwse-resize}   
zoom-in P {cursor: zoom-in} IE не поддерживает.  
zoom-out P {cursor: zoom-out} IE не поддерживает.  
grab P {cursor: grab} Chrome, Opera, Safari поддерживают значение -  
 webkit-grab. IE не поддерживает.  
grabbing P {cursor: grabbing} Chrome, Opera, Safari поддерживают значение   
 webkit-grabbing. IE не поддерживает.

В зависимости от операционной системы и её настроек вид курсора может отличаться от приведённых в таблице. При добавлении курсора из файла синтаксис несколько видоизменится.

cursor: url(<адрес>), url(<адрес>), ..., <курсор>

Через запятую допускается указывать несколько значений url, в этом случае браузер попытается открыть первый файл с курсором и если это по каким-либо причинам не получится, перейдёт к следующему файлу. Список обязательно заканчивается ключевым словом, например, auto или pointer, допустимые значения перечислены выше.

***.cross { cursor: crosshair; }  
 .help { cursor: help; }***

***<p class="cross">На этом тексте курсор мыши примет вид перекрестья.</p>  
 <p>   
 <a href="page/help.html" class="help">СПРАВКА 1</a><br />  
 <a href="page/help.html" class="help">СПРАВКА 2</a><br />  
 <a href="page/help.html" class="help">СПРАВКА 3</a>  
 </p>***