**Урок 5**

Эффекты перехода и трансформации CSS3

Создание анимации средствами CSS3. Технология рисования canvas. Оживим страницы нашего сайта.

[CSS3. Трансформация](#_ie0madmg9jzy)

[transform](#_8hr0wsfphr42)

[Практика](#_gfbim8m6pjss)

[Поворот](#_30j0zll)

[Масштабирование](#_82l5foibgg5k)

[Перемещение](#_bki29f13swj2)

[Искажение](#_5hg03lwzvriq)

[Использование нескольких трансформаций](#_c55chfrgdofz)

[Точка трансформации transform-origin](#_t3264wq22y30)

[Множественные трансформации](#_v3p9n6q11qhf)

[CSS3 transition](#_pmdpudoyhfsh)

[Практика](#_gfrsxu8d837k)

[Переходы](#_6jdj0291oi7b)

[Простой цветовой переход](#_6rjyecip21xj)

[Задержка и комбинирование эффектов перехода](#_uloz7wunjomw)

[CSS3 Анимация](#_93rs70mkig4y)

[Ход выполнения анимации](#_1fob9te)

[Длительность анимации animation-duration](#_3znysh7)

[Временная функция animation-timing-function](#_2et92p0)

[Шаги анимации](#_tyjcwt)

[Анимация с задержкой animation-delay](#_3dy6vkm)

[Повтор анимации animation-iteration-count](#_1t3h5sf)

[Направление анимации animation-direction](#_4d34og8)

[Проигрывание анимации animation-play-state](#_2s8eyo1)

[Состояние элемента до и после воспроизведения анимации animation-fill-mode](#_17dp8vu)

[Краткая запись анимации](#_3rdcrjn)

[Множественные анимации](#_26in1rg)

[Практическое задание](#_z337ya)

[Дополнительные материалы](#_3j2qqm3)

[Используемая литература](#_uvp6qax5r1ok)

# CSS3. Трансформация

## transform

Свойство задает вид преобразования элемента. Описывается с помощью функций трансформации, которые смещают элемент относительно его текущего положения на странице или изменяют его первоначальные размеры и форму. Не наследуется.

Допустимые значения:

* matrix() – любое число.
* translate(), translateX(), translateY() – единицы длины (положительные и отрицательные), %.
* scale(), scaleX(), scaleY() – любое число.
* rotate() – угол (deg, grad, rad или turn).
* skew(), skewX(), skewY() – угол (deg, grad, rad).

1. matrix(a, c, b, d, x, y)
   1. Смещает элементы и задает способ их трансформации, позволяя объединить несколько функций 2D-трансформаций в одной. В качестве трансформации допустимы поворот, масштабирование, наклон и изменение положения. Значение a изменяет масштаб по горизонтали.
   2. Значение от 0 до 1 уменьшает элемент, больше 1– увеличивает.
   3. Значение c деформирует (сдвигает) стороны элемента по оси Y, положительное значение – вверх, отрицательное – вниз.
   4. Значение b деформирует (сдвигает) стороны элемента по оси X, положительное значение – влево, отрицательное – вправо.
   5. Значение d изменяет масштаб по вертикали. Значение меньше 1 уменьшает элемент, больше 1 – увеличивает.
   6. Значение x смещает элемент по оси X, положительное – вправо, отрицательное – влево.
   7. Значение y смещает элемент по оси Y, положительное значение – вниз, отрицательное – вверх.
2. translate(x,y)
   1. Сдвигает элемент на новое место, перемещая относительно обычного положения вправо и вниз, используя координаты x и y, не затрагивая при этом соседние элементы. Если нужно сдвинуть элемент влево или вверх, нужно использовать отрицательные значения.
3. translateX(n)
   1. Сдвигает элемент относительно его обычного положения по оси X.
4. translateY(n)
   1. Сдвигает элемент относительно его обычного положения по оси Y.
5. scale(x,y)
   1. Масштабирует элементы, делая их больше или меньше. Значения от 0 до 1 уменьшают элемент. Первое значение масштабирует элемент по ширине, второе – по высоте. Отрицательные значения отображают элемент зеркально.
   2. scaleX(n) Функция масштабирует элемент по ширине, делая его шире или уже. Если значение больше единицы, элемент становится шире, если значение находится между единицей и нулем, элемент становится уже. Отрицательные значения отображают элемент зеркально по горизонтали.
   3. scaleY(n) Функция масштабирует элемент по высоте, делая его выше или ниже. Если значение больше единицы, элемент становится ниже, если значение находится между единицей и нулем – ниже. Отрицательные значения отображают элемент зеркально по вертикали.
6. rotate(угол)
   1. Поворачивает элементы на заданное количество градусов, отрицательные значения от -1deg до -360deg поворачивают элемент против часовой стрелки, положительные – по часовой стрелке. Значение rotate(720deg) поворачивает элемент на два полных оборота.
7. skew(x-угол,y-угол)
   1. Используется для деформирования (искажения) сторон элемента относительно координатных осей. Если указано одно значение, второе будет определено браузером автоматически.
   2. skewX(угол) Деформирует стороны элемента относительно оси X.
   3. skewY(угол) Деформирует стороны элемента относительно оси Y.
   4. initial Устанавливает значение свойства в значение по умолчанию.
   5. inherit Наследует значение свойства от родительского элемента.

# Практика

## Поворот

Трансформации CSS3 являются новой и экспериментальной возможностью. Поэтому при их использовании нужно указывать несколько версий свойства transform, каждую со своим префиксом разработчика браузеров. Далее приведен пример правила трансформации для вращения элемента со всем его содержимым:

|  |
| --- |
| <!-- HTML-разметка трансформируемого элемента --> <figure>  <img src="/image/7.png" />  <figcaption>Рисунок</figcaption>  </figure> |

|  |
| --- |
| figure {  -moz-transform: rotate(45deg);  -webkit-transform: rotate(45deg);  -o-transform: rotate(45deg);  -ms-transform: rotate(45deg);  transform: rotate(45deg); } |

Применение трансформации к какому-либо элементу страницы не влияет на другие ее элементы или ее компоновку. Например, если повернуть элемент, он просто надвинется на смежные с ним элементы.

Обратите внимание, что в предыдущем примере угол поворота задавался в градусах. Спецификация CSS3 предлагает возможность задавать угол и в других математических величинах.

Единицы измерения угла поворота CSS3-трансформации:

|  |  |  |  |
| --- | --- | --- | --- |
| **Единица измерения** | **CSS3-**  **обозначение** | **Описание** | **Пример** |
| Градусы | *deg* | Угол полной окружности 360° | rotate(90deg) |
| Грады | *grad* | Единица измерения плоских углов, равная 1/400 угла полной окружности, или π/200 | rotate(100grad) |
| Радианы | *rad* | 2π радиан равно углу полной окружности | rotate(1.57rad) |
| Обороты | *turn* | 1 угол полной окружности равен одному обороту | rotate(.25turn) |

## Масштабирование

Для использования анимации масштабирования используется функция scale(). Значение масштаба задается относительно единицы: scale(2) – исходный элемент будет увеличен в два раза, scale(0.5) – элемент уменьшится в два раза. Согласно этим правилам элемент масштабируется одинаково во все стороны, но можно задать направление: X – по горизонтали, Y – по вертикали, Z – глубина масштабирования. Более того, функцию scale() можно использовать для создания эффекта отражения. Для этого нужно передать в нее отрицательное значение.

Рассмотрим некоторые примеры масштабирования. Для начала уменьшим картинку с текстом в два раза:

|  |
| --- |
| figure {  -moz-transform: scale(0.5);  -webkit-transform: scale(0.5);  -o-transform: scale(0.5);  -ms-transform: scale(0.5);  transform: scale(0.5); } |

Растянем по оси Х в два раза:

|  |
| --- |
| figure {  -moz-transform: scaleX(2);  -webkit-transform: scaleX(2);  -o-transform: scaleX(2);  -ms-transform: scaleX(2);  transform: scaleX(2); } |

Используем эффект отражения:

|  |
| --- |
| figure {  -moz-transform: scaleX(-1);  -webkit-transform: scaleX(-1);  -o-transform: scaleX(-1);  -ms-transform: scaleX(-1);  transform: scaleX(-1); } |

## Перемещение

Для перемещения HTML-элемента используется функция translate(x,y) либо ее аналоги для конкретных осей translateX(x), translateY(y). Эти функции поддерживают отрицательные значения (сдвиг влево или вверх). Ниже показан пример использования трансформации перемещения:

|  |
| --- |
| figure {  -moz-transform: translate(50px, -4em);  -webkit-transform: translate(50px, -4em);  -o-transform: translate(50px, -4em);  -ms-transform: translate(50px, -4em);  transform: translate(50px, -4em); } |

## Искажение

Трансформация skew() искажает форму элемента. Например, возьмем правильный прямоугольник с закрепленным основанием. Если мы начнем толкать его верхнюю часть в сторону, она сместится, в то время как основание останется на месте. В результате получим параллелограмм.

Значение искажения задается в градусах (можно указывать такие же единицы измерения, как для трансформации поворота). Ниже показан пример использования трансформации искажения:

|  |
| --- |
| figure {  -moz-transform: skewX(50grad);  -webkit-transform: skewX(50grad);  -o-transform: skewX(50grad);  -ms-transform: skewX(50grad);  transform: skewX(50grad); } |

Следующий пример показывает искажение в градусах сразу по двум осям:

|  |
| --- |
| figure {  transform:skew(40deg,20deg);  -webkit-transform:skew(40deg,20deg); *// для Chrome и Safari*  -ms-transform:skew(40deg,20deg); *// для IE* } |

## Использование нескольких трансформаций

Вы можете объединить несколько трансформаций: указать их в одном правиле, разделив пробелами. Ниже приведен пример применения сразу нескольких трансформаций к одному элементу.

|  |
| --- |
| figure {  -moz-transform: scale(1.5) translateX(10px) skew(10deg) rotate(0.175rad);  -webkit-transform: scale(1.5) translateX(10px) skew(10deg) rotate(0.175rad);  -o-transform: scale(1.5) translateX(10px) skew(10deg) rotate(0.175rad);  -ms-transform: scale(1.5) translateX(10px) skew(10deg) rotate(0.175rad);  transform: scale(1.5) translateX(10px) skew(10deg) rotate(0.175rad); } |

Сначала элемент увеличивается в полтора раза (используя трансформацию scale), потом перемещается на 10 пикселей влево (посредством трансформации translateX) и, наконец, наклоняется и поворачивается (используя трансформации skew и rotate).

## Точка трансформации transform-origin

Свойство позволяет сместить центр трансформации, относительно которого происходит изменение положения/размера/формы элемента. Значение по умолчанию – center, или 50% 50%. Задается только для трансформированных элементов. Не наследуется.

## Множественные трансформации

Можно объединить несколько трансформаций одного элемента, перечислив их через пробел в порядке проявления.

# CSS3 transition

При добавлении плавности переходов для элементов для общего свойства transition существуют его составляющие (аналогично свойству background, background-color, background-image и т.д.).

Рассмотрим, из чего состоит общее свойство transition:

1. Свойство transition-property. Содержит название CSS-свойств, к которым будет применен эффект перехода. Не наследуется.
   1. Значения:
      1. **none** – отсутствие свойства для перехода.
      2. **all** –значение по умолчанию. Применяет эффект перехода ко всем свойствам элемента.
      3. **свойство** – определяет список CSS-свойств, перечисленных через запятую, участвующих в переходе.
2. Продолжительность перехода transition-duration. Задает промежуток времени, в течение которого должен осуществляться переход. Не наследуется.
   1. Значения:
      1. **время** перехода указывается в секундах или миллисекундах, например, 1s, или 5ms, или 0.3s.
3. Функция перехода transition-timing-function. Свойство задает временную функцию, которая определяет скорость перехода объекта от одного значения к другому. Если вы определяете более одного перехода для элемента, например цвет фона элемента и его положение, вы можете перечислить нужные переходы через запятую.
   1. Значения:
      1. **ease** – функция по умолчанию, переход начинается медленно, разгоняется быстро и замедляется в конце. Соответствует cubic-bezier(0.25,0.1,0.25,1).
      2. **linear** – переход происходит равномерно на протяжении всего времени, без колебаний в скорости. Соответствует cubic-bezier(0,0,1,1).
      3. **ease-in** – переход начинается медленно, а затем плавно ускоряется в конце. Соответствует cubic-bezier(0.42,0,1,1).
      4. **ease-out** – переход начинается быстро и плавно замедляется в конце. Соответствует cubic-bezier(0,0,0.58,1).
      5. **ease-in-out** – переход медленно начинается и медленно заканчивается. Соответствует cubic-bezier(0.42,0,0.58,1).
      6. **cubic-bezier(x1, y1, x2, y2)** – позволяет вручную установить значения от 0 до 1 для кривой ускорения.
4. Задержка перехода transition-delay. Необязательное свойство, позволяет сделать так, чтобы изменение свойства происходило не моментально, а с некоторой задержкой. Не наследуется.
   1. **Время** задержки перехода указывается в секундах или миллисекундах.
5. Краткая запись перехода transition. Все свойства, отвечающие за изменение внешнего вида элемента, можно объединить в одно свойство transition.

# Практика

## Переходы

С помощью псевдоклассов :hover и :focus можно создавать интерактивные эффекты, не прибегая к использованию сценариев JavaScript. Например, чтобы создать меняющуюся кнопку (т.е. кнопку, реагирующую на наведение курсора мыши), достаточно просто предоставить набор новых свойств стиля для псевдокласса :hover. Эти свойства задействуются автоматически, когда пользователь наводит курсор на кнопку.

CSS3 предоставляет более простой способ переходов, позволяющий осуществлять плавный переход от одного набора стилей к другому.

## Простой цветовой переход

Чтобы понять принцип работы переходов, рассмотрим пример. Изменение цвета кнопки при наведении на нее курсора осуществляется с помощью возможностей перехода CSS3.

При наведении курсора ее цвет меняется с зеленого на желтый резким прыжком. Но при использовании эффекта перехода зеленый цвет переходит в желтый плавно. Таким же образом происходит и обратный переход при отодвигании курсора.

Смену цвета без эффекта перехода можно реализовать следующим кодом:

|  |
| --- |
| .slickButton {  color: white;  font-weight: bold;  padding: 10px;  margin: 20px;  border: solid 2px gray;  background: lightgreen;  cursor: pointer; } .slickButton:hover {  color: black;  background: yellow; } |

Чтобы получить плавное изменение цвета, т.е. переход, нам нужно в только что описанный стиль добавить свойство transition. Обратите внимание, что это свойство вставляется в обычный стиль (в данном случае стиль slickButton), а не в псевдокласс :hover.

Как минимум, свойство transition требует установки двух значений: свойства CSS, которое нужно анимировать, и времени, на протяжении которого нужно выполнить изменение стилей. В данном примере переход применяется к свойству background, а время перехода равно 0.5 секунды:

|  |
| --- |
| .slickButton {  */\* ... \*/*  -webkit-transition: background 0.5s, color 0.5s;  -moz-transition: background 0.5s, color 0.5s;  -o-transition: background 0.5s, color 0.5s; } |

Этот код можно сократить, если нам нужно установить переход для всех изменяющихся свойств и при одинаковом времени перехода для всех. В таком случае вместо списка свойств для перехода мы просто используем ключевое слово all:

|  |
| --- |
| .slickButton {  */\* ... \*/*  -webkit-transition: all 0.5s;  -moz-transition: all 0.5s;  -o-transition: all 0.5s; } |

## 

## Задержка и комбинирование эффектов перехода

В эффектах перехода можно использовать свойство transition-delay, которое задерживает начало перехода на указанное время:

|  |
| --- |
| .slickButton {  */\* ... \*/*  -webkit-transition: all 0.5s;  -moz-transition: all 0.5s;  -o-transition: all 0.5s;  transition: all 0.5s;  transition-delay: 2s; } |

Теперь анимация перехода при наведении указателя мыши будет задерживаться на две секунды. То же самое произойдет и при отведении указателя мыши. Значение свойства transition-delay можно указывать в самом свойстве transition как необязательный параметр:

|  |
| --- |
| .slickButton {  */\* ... \*/*  -webkit-transition: all 0.5s 2s;  -moz-transition: all 0.5s 2s;  -o-transition: all 0.5s 2s;  transition: all 0.5s 2s; } |

# CSS3 Анимация

Для браузеров Chrome и Safari перед свойством требуется добавить префикс -webkit.

Для создания анимации в CSS3 используется свойство @keyframes.

Оно представляет собой контейнер, в который должны помещаться различные свойства оформления.

|  |
| --- |
| @keyframes имяАнимации { from {CSS свойства} */\* Оформление элемента перед началом анимации \*/* to {CSS свойства} */\* Оформление элемента после завершения анимации \*/* } |

После того, как анимация была создана, необходимо добавить к элементу, который Вы хотите анимировать, CSS3 свойство animation и указать в нем имя анимации (1 значение) и время (2 значение), в течение которого она будет выполняться.

Также Вы можете устанавливать количество повторов анимации (3 значение).

|  |
| --- |
| @keyframes anim { from {margin-left:3px;} to {margin-left:500px;} } #wrap1 { animation:anim 4s 3; } |

Полный листинг анимированной страницы представлен ниже.

|  |
| --- |
| <html> <style type='text/css'> @keyframes anim{ from {margin-left:3px;} to {margin-left:500px;} } @-moz-keyframes anim{ from {margin-left:3px;} to {margin-left:500px;} } @-webkit-keyframes anim{ from {margin-left:3px;} to {margin-left:500px;} } #wrap1{ border:2px #000 solid; background-color:#7F0055; height:100px; width:100px; font-size:2em; animation:anim 4s 3; -webkit-animation:anim 4s 3; } </style> <body> <div id="wrap1"></div> <p><b>Обратите внимание:</b> данная анимация будет повторяться 3 раза.</p> </body> </html> |

## Ход выполнения анимации

Вы можете определять ход выполнения анимации не только с помощью ключевых слов from и to (которые использовались в предыдущем примере), но и с помощью %.

С помощью % Вы можете более точно контролировать ход выполнения анимации – например, указать, что определенный элемент в начале анимации (0%) должен быть белым, к середине (50%) должен окрашиваться в оранжевый цвет, а к концу (100%) – становиться черным.

|  |
| --- |
| @keyframes anim { 0% {margin-left:3px;margin-top:3px;background-color:#7F0055;} 30% {margin-left:3px;margin-top:250px;background-color:#7F0055;} 60% {margin-left:500px;margin-top:250px;background-color:black;} 100% {margin-left:3px;margin-top:3px;background-color:#7F0055;} } #wrap1 { animation:anim 6s 3; } |

Полный листинг анимированной страницы представлен ниже:

|  |
| --- |
| <html> <style type='text/css'> @keyframes anim { 0% {margin-left:3px;margin-top:3px;background-color:#7F0055;} 30% {margin-left:3px;margin-top:250px;background-color:#7F0055;} 60% {margin-left:500px;margin-top:250px;background-color:black;} 100% {margin-left:3px;margin-top:3px;background-color:#7F0055;} } @-moz-keyframes anim { 0% {margin-left:3px;margin-top:3px;background-color:#7F0055;} 30% {margin-left:3px;margin-top:250px;background-color:#7F0055;} 60% {margin-left:500px;margin-top:250px;background-color:black;} 100% {margin-left:3px;margin-top:3px;background-color:#7F0055;} } @-webkit-keyframes anim { 0% {margin-left:3px;margin-top:3px;background-color:#7F0055;} 30% {margin-left:3px;margin-top:250px;background-color:#7F0055;} 60% {margin-left:500px;margin-top:250px;background-color:black;} 100% {margin-left:3px;margin-top:3px;background-color:#7F0055;} } #wrap1 { border:2px #000 solid; background-color:#7F0055; height:100px; width:100px; font-size:2em; animation:anim 6s 3; -webkit-animation:anim 6s 3; } </style> <body> <div id="wrap1"></div> </body> </html> |

CSS3 свойства анимации

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| @keyframes | Контейнер для определения анимации |
| animation | Позволяет задать все значения для настройки выполнения анимации за одно определение |
| animation-name | Позволяет указать имя анимации |
| animation-duration | Позволяет задать скорость выполнения анимации в секундах (*по умолчанию имеет значение 0*) |
| animation-timing-function | Позволяет задать функцию смягчения, отвечающую за плавность выполнения анимации (*по умолчанию имеет значение ease*) |
| animation-delay | Позволяет задать задержку перед началом выполнения анимации (*по умолчанию имеет значение 0*) |
| animation-iteration-count | Позволяет задать количество повторов анимации (*по умолчанию имеет значение 1*) |
| animation-direction | При значении alternate в нечетные разы (*1, 3, 5 ...*) анимация будет проигрываться в нормальном, а в четные (*2, 4, 6 ...*) – в обратном порядке. По умолчанию данное свойство имеет значение normal, при данном значении анимация всегда проигрывается в нормальном порядке. |

## Длительность анимации animation-duration

Свойство устанавливает длительность анимации. Не наследуется. Значение по умолчанию 0.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| время | Длительность анимации задается в секундах s или миллисекундах ms |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

Синтаксис

|  |
| --- |
| -webkit-animation-duration: 2s; animation-duration: 2s; |

## 

## Временная функция animation-timing-function

Свойство определяет изменение скорости от начала до конца анимации с помощью временных функций. Задается при помощи ключевых слов или кривой Безье cubic-bezier(x1, y1, x2, y2). Не наследуется.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| ease | Функция по умолчанию, анимация начинается медленно, разгоняется быстро и замедляется в конце. Соответствует cubic-bezier(0.25,0.1,0.25,1) |
| linear | Анимация происходит равномерно на протяжении всего времени, без колебаний в скорости. Соответствует cubic-bezier(0,0,1,1) |
| ease-in | Анимация начинается медленно, а затем плавно ускоряется в конце. Соответствует cubic-bezier(0.42,0,1,1) |
| ease-out | Анимация начинается быстро и плавно замедляется в конце. Соответствует cubic-bezier(0,0,0.58,1) |
| ease-in-out | Анимация медленно начинается и медленно заканчивается. Соответствует cubic-bezier(0.42,0,0.58,1) |
| cubic-bezier(x1, y1, x2, y2) | Позволяет вручную установить значения от 0 до 1. [На этом сайте](http://roblaplaca.com/examples/bezierBuilder/) вы сможете построить любую траекторию скорости изменения анимации |
| step-start | Задает пошаговую анимацию, разбивая ее на отрезки, изменения происходят в начале каждого шага. Эквивалентно steps(1, start) |
| step-end | Пошаговая анимация, изменения происходят в конце каждого шага. Эквивалентно steps(1, end) |
| steps(количество шагов,start|end) | Ступенчатая временная функция, которая принимает два параметра. Количество шагов задается целым положительным числом. Второй параметр необязательный, указывает момент, в котором начинается анимация. Со значением start анимация начинается в начале каждого шага, со значением end – в конце каждого шага с задержкой. Задержка вычисляется как результат деления времени анимации на количество шагов. Если второй параметр не указан, используется значение по умолчанию end |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

## Шаги анимации

Элемент можно анимировать используя шаги, т.е. ступенчато. (Примером может послужить секундная стрелка часов, которая сначала двигается, а затем осуществляется задержка на 1 секунду, потом снова двигается и снова задержка и т.д.) Шаги задаются с помощью функции steps(). Ниже показан пример:

|  |
| --- |
| transition: all 2000ms steps(3, end); |

Теперь анимация увеличения кнопки будет происходить рывками. Задержка между рывками в данном случае будет 667 ms (2000/3). Второй параметр функции steps указывает, будет рывок выполняться сразу или после задержки.

## Анимация с задержкой animation-delay

Свойство игнорирует анимацию заданное количество времени, что дает возможность по отдельности запускать каждую анимацию. Отрицательная задержка начинает анимацию с определенного момента внутри ее цикла, т.е. со времени, указанного в задержке. Это позволяет применять анимацию к нескольким элементам со сдвигом фазы, изменяя лишь время задержки.

Чтобы анимация началась с середины, нужно задать отрицательную задержку, равную половине времени, установленного в animation-duration. Не наследуется.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| время | Задержка анимации задается в секундах s или миллисекундах ms.  Значение по умолчанию 0 |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

Синтаксис:

|  |
| --- |
| -webkit-animation-delay: 2s;  animation-delay: 2s; |

## Повтор анимации animation-iteration-count

Свойство позволяет запустить анимацию несколько раз. Значение 0 или любое отрицательное число удаляют анимацию из проигрывания. Не наследуется.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| число | С помощью целого числа задается количество повторов анимации. Значение по умолчанию 1 |
| infinite | Анимация проигрывается бесконечно |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

Синтаксис:

|  |
| --- |
| -webkit-animation-iteration-count: 3;  animation-iteration-count: 3; |

## 

## Направление анимации animation-direction

Свойство задает направление повтора анимации. Если анимация повторяется только один раз, это свойство не имеет смысла. Не наследуется.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| alternate | Анимация проигрывается с начала до конца, затем в обратном направлении |
| alternate-reverse | Анимация проигрывается с конца до начала, затем в обратном направлении |
| normal | Значение по умолчанию, анимация проигрывается в обычном направлении, с начала и до конца |
| reverse | Анимация проигрывается с конца |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

Синтаксис

|  |
| --- |
| -webkit-animation-direction: alternate;  animation-direction: alternate; |

## Проигрывание анимации animation-play-state

Свойство управляет проигрыванием и остановкой анимации. Остановка анимации внутри цикла возможна через использование этого свойства в скрипте JavaScript. Также можно останавливать анимацию при наведении курсора мыши на объект – состояние :hover. Не наследуется.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| paused | Останавливает анимацию |
| running | Значение по умолчанию, означает проигрывание анимации |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

Синтаксис:

|  |
| --- |
| -webkit-animation-play-state: paused; animation-play-state: paused; |

## Состояние элемента до и после воспроизведения анимации animation-fill-mode

Свойство определяет порядок применения определенных в @keyframes стилей к объекту. Не наследуется.

|  |  |
| --- | --- |
| **Значения** | **Описание** |
| none | Значение по умолчанию. Состояние элемента не меняется до или после воспроизведения анимации |
| forwards | Воспроизводит анимацию до последнего кадра по окончанию последнего повтора и не отматывает ее к первоначальному состоянию |
| backwards | Возвращает состояние элемента после загрузки страницы к первому кадру, даже если установлена задержка animation-delay, и оставляет его там, пока не начнется анимация |
| both | Позволяет оставлять элемент в первом ключевом кадре до начала анимации (игнорируя положительное значение задержки) и задерживать на последнем кадре до конца последней анимации |
| initial | Устанавливает значение свойства в значение по умолчанию |
| inherit | Наследует значение свойства от родительского элемента |

Синтаксис:

|  |
| --- |
| -webkit-animation-fill-mode: forwards;  animation-fill-mode: forwards; |

## Краткая запись анимации

Все параметры воспроизведения анимации можно объединить в одном свойстве – animation, перечислив их через пробел:

|  |
| --- |
| animation: animation-name animation-duration animation-timing-function animation-delay animation-iteration-count animation-direction; |

## Множественные анимации

Для одного элемента можно задавать несколько анимаций, перечислив их названия через запятую.

Синтаксис:

|  |
| --- |
| div {animation: shadow 1s ease-in-out 0.5s alternate, move 5s linear 2s;} |

# Практическое задание

1. Создать страницу checkout.psd.
2. Применить к нескольким элементам на странице эффекты:
   1. Трансформации (увеличения изображения при наведении).
   2. Поворота (добавить поворот элемента на 180 или на 360 градусов при наведении на него).
   3. Сделать трансформацию элементов плавной (transition).
   4. \* Анимации (по желанию).
   5. \* Применить к произвольному элементу технологию рисования canvas.
3. \* Для некоторых элементов применить функцию, которая позволяет изменять скорость перехода в процессе его осуществления.
4. \* Использовать для некоторых элементов множественную анимацию.

# Дополнительные материалы

1. <https://webref.ru/css/transition-timing-function>.
2. <https://developer.mozilla.org/en-US/docs/Web/CSS/transition-property>.

# Используемая литература

Для подготовки данного методического пособия были использованы следующие ресурсы:

1. <http://www.wisdomweb.ru/>.
2. <http://html5book.ru/css3-transform/>.
3. Гоше Х. HTML5. Для профессионалов. – СПб.: Питер, 2013. – 496 с.: ил.
4. Хоган Б. HTML5 и CSS3. Веб-разработка по стандартам нового поколения. – СПб.: Питер, 2012. – 272 с.
5. Макфарланд Д. Большая книга CSS3. – СПб.: Питер, 2016. – 608 с.