

CSS3: переходы, трансформация, анимация

CSS3: переходы

CSS3-переходы позволяют анимировать исходное значение CSS-свойства на новое значение с течением времени, управляя скоростью смены значений свойств.

Для задания всех свойств перехода обычно используют краткую запись свойства **transition**.

```
div {  
  width: 100px;  
  transition-property: width;  
}  
div:hover {  
  width: 300px;  
}
```

Большинство свойств меняют свои значения за 16 миллисекунд, поэтому рекомендуемое время стандартного перехода — 200ms.

Свойство **transition-property** содержит название CSS-свойств, к которым будет применен эффект перехода. Значение свойства может содержать как одно свойство, так и список свойств через запятую.

Продолжительность перехода: transition-duration

Задаёт промежуток времени, в течение которого должен осуществляться переход.

- ✓ Если разные свойства имеют разные значения для перехода, они указываются через запятую.
- ✓ Если продолжительность перехода не указана, то анимация при смене значений свойств происходить не будет.

```
div {  
  transition-duration: .2s;  
}
```

transition-duration	
время	Время перехода указывается в секундах или миллисекундах, например, 2s или 5ms.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Функция перехода: transition-timing-function

Свойство задаёт временную функцию, которая описывает скорость перехода объекта от одного значения к другому.

transition-duration	
ease	Функция по умолчанию, переход начинается медленно, разгоняется быстро и замедляется в конце.
linear	Переход происходит равномерно на протяжении всего времени, без колебаний в скорости.
ease-in	Переход происходит равномерно на протяжении всего времени, без колебаний в скорости.
ease-out	Переход происходит равномерно на протяжении всего времени, без колебаний в скорости.
ease-in-out	Переход медленно начинается и медленно заканчивается
cubic-bezier(x1, y1, x2, y2)	Позволяет вручную установить значения от 0 до 1 для кривой ускорения
Initial, inherit	

Задержка перехода: transition-delay

Необязательное свойство, позволяет сделать так, чтобы изменение свойства происходило не моментально, а с некоторой задержкой.

Не наследуется.

```
div {  
  transition-delay: .5s;  
}
```

transition-duration	
время	Время задержки перехода указывается в секундах или миллисекундах
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

CSS3: переходы

Краткая запись перехода

Все свойства, отвечающие за изменение внешнего вида элемента, можно объединить в одно свойство transition

transition: transition-property transition-duration
transition-timing-function transition-delay;

```
div {transition: 1s;}
```



```
div {transition: all 1s ease 0s;}
```

Плавный переход нескольких свойств

Для элемента можно задать несколько последовательных переходов, перечислив их через запятую. Каждый переход можно оформить своей временной функцией.

```
div {transition: background 0.3s ease, color 0.2s linear;}
```

или

```
div {  
  transition-property: height, width, background-color;  
  transition-duration: 3s;  
  transition-timing-function: ease-in, ease, linear;  
}
```

CSS3: трансформация

CSS3-трансформации позволяют сдвигать, поворачивать и масштабировать элементы. Трансформации преобразовывают элемент, не затрагивая остальные элементы веб-страницы, т.е. другие элементы не сдвигаются относительно него.

Свойство **transform** задаёт вид преобразования элемента.

Свойство описывается с помощью **функций трансформации**, которые смещают элемент относительно его текущего положения на странице или изменяют его первоначальные размеры и форму.

`matrix()` — любое число

`translate()`, `translateX()`, `translateY()` — единицы длины

отрицательные), `%`

`scale()`, `scaleX()`, `scaleY()` — любое число

`rotate()` — угол (`deg`, `grad`, `rad` или `turn`)

`skew()`, `skewX()`, `skewY()` — угол (`deg`, `grad`, `rad`)

Точка трансформации: transform-origin

Свойство **transform-origin** позволяет сместить центр трансформации, относительно которого происходит изменение положения/размера/формы элемента.

```
transform-origin: 2px;  
transform-origin: bottom;  
transform-origin: 3cm 2px;  
transform-origin: left 2px;  
transform-origin: right top;  
transform-origin: inherit;  
transform-origin: initial;
```

transform-origin

**ось X(left, center, right,
длина, %)**

ось

**Y(top, center, bottom,
длина, %)**

Время задержки перехода
указывается в секундах или
миллисекундах

initial

Устанавливает значение
свойства в значение по
умолчанию.

inherit

Наследует значение
свойства от родительского
элемента.

Точка трансформации: transform-origin

Свойство **transform-origin** позволяет сместить центр трансформации, относительно которого происходит изменение положения/размера/формы элемента.

```
transform-origin: 2px;  
transform-origin: bottom;  
transform-origin: 3cm 2px;  
transform-origin: left 2px;  
transform-origin: right top;  
transform-origin: inherit;  
transform-origin: initial;
```

transform-origin



transform-origin



Ключевые кадры

Ключевые кадры используются для указания значений свойств анимации в различных точках анимации.

Ключевые кадры определяют поведение одного цикла анимации; анимация может повторяться ноль или более раз.

Ключевые кадры указываются с помощью правила **@keyframes**, определяемого следующим образом:

```
}  
@keyframes loader {  
  0% {transform:rotate(-45deg)}  
  50%{transform:rotate(-135deg)}  
  100%{transform:rotate(-225deg)}}  
@keyframes span-1 {  
  0% {transform:translate(0);}   
  50%{transform:translate(-50px, 0);border-  
color:#EE4D68}  
  100%{transform:translate(0);}  
@keyframes span-2 {  
  0%{transform:translate(0);}
```



```
@keyframes имя анимации { список правил }
```

CSS3: анимация

CSS3-анимация может применяться практически для всех html-элементов, а также для псевдоэлементов `:before` и `:after`.

Создание анимации базируется на **ключевых кадрах**, которые позволяют автоматически воспроизводить и повторять эффекты на протяжении заданного времени, а также останавливать анимацию внутри цикла.

```
}  
@keyframes loader {  
  0% {transform:rotate(-45deg)}  
  50%{transform:rotate(-135deg)}  
  100%{transform:rotate(-225deg)}}  
@keyframes span-1 {  
  0% {transform:translate(0);}   
  50%{transform:translate(-50px, 0);border-color:#EE4D68}  
  100%{transform:translate(0);}}  
@keyframes span-2 {  
  0%{transform:translate(0);}
```



Ключевые кадры

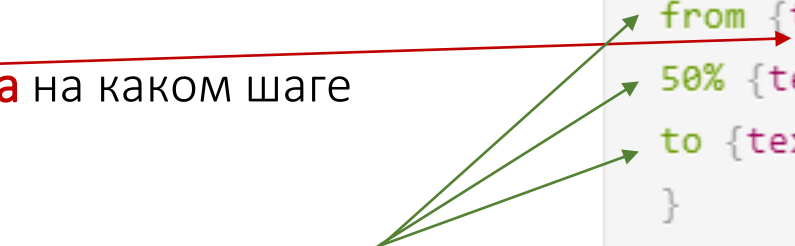
Создание анимации начинается с установки **ключевых кадров** правила `@keyframes`.

Кадры определяют, **какие свойства** на каком шаге будут анимированы.

Каждый кадр может включать **один или более блоков** объявления из одного или более пар свойств и значений.

Правило `@keyframes` содержит имя анимации элемента, которое связывает правило и блок объявления элемента.

```
@keyframes shadow {  
  from {text-shadow: 0 0 3px black;}  
  50% {text-shadow: 0 0 30px black;}  
  to {text-shadow: 0 0 3px black;}  
}
```



Ключевые кадры

После объявления правила **@keyframes**, можно ссылаться на него в свойстве **animation**:



Не рекомендуется анимировать нечисловые значения (за редким исключением), так как результат в браузере может быть непредсказуемым.

Не следует создавать ключевые кадры для значений свойств, не имеющих средней точки:

color: pink и color: #ffffff, width: auto и width: 100px или border-radius: 0 и border-radius: 50% (в этом случае правильно будет указать border-radius: 0%).

```
@keyframes shadow {  
  from {text-shadow: 0 0 3px black;}  
  50% {text-shadow: 0 0 30px black;}  
  to {text-shadow: 0 0 3px black;}  
}
```

```
h1 {  
  font-size: 3.5em;  
  color: darkmagenta;  
  animation: shadow 2s infinite ease-in-out;  
}
```

Название анимации: свойство **animation-name**

Свойство **animation-name** определяет список применяемых к элементу анимаций.

- ✓ Имя анимации чувствительно к регистру,
- ✓ не допускается использование ключевого слова none,
- ✓ рекомендуется использовать название, отражающее суть анимации,
- ✓ можно использовать одно или несколько слов, перечисленных через дефис - или символ нижнего подчеркивания _.

```
animation-name: none;  
animation-name: test-01;  
animation-name: -sliding;  
animation-name: moving-vertically;  
animation-name: test2;  
animation-name: test3, move4;  
animation-name: initial;  
animation-name: inherit;
```



Свойство не наследуется.

Название анимации: свойство `animation-name`

<code>animation-name</code>	
Значения:	
<code>none</code>	Означает отсутствие анимации. Также используется, чтобы отменить анимацию элемента из группы элементов, для которых задана анимация. Значение по умолчанию.
имя анимации	Имя анимации, которое связывает правило <code>@keyframes</code> с селектором.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
animation-name: none;  
animation-name: test-01;  
animation-name: -sliding;  
animation-name: moving-vertically;  
animation-name: test2;  
animation-name: test3, move4;  
animation-name: initial;  
animation-name: inherit;
```



Свойство не наследуется.

Продолжительность анимации: свойство `animation-duration`

Свойство `animation-duration` определяет продолжительность одного цикла анимации.

- ✓ Задаётся в секундах `s` или миллисекундах `ms`.
- ✓ Если для элемента задано более одной анимации, то можно установить разное время для каждой, перечислив значения через запятую.

```
animation-duration: .5s;  
animation-duration: 200ms;  
animation-duration: 2s, 10s;  
animation-duration: 15s, 30s, 200ms;
```



Свойство не наследуется.

Временная функция: свойство `animation-timing-function`

Свойство `animation-timing-function` описывает, как будет развиваться анимация между каждой парой ключевых кадров.

```
animation-timing-function: ease;  
animation-timing-function: ease-in;  
animation-timing-function: ease-out;  
animation-timing-function: ease-in-out;  
animation-timing-function: linear;  
animation-timing-function: step-start;  
animation-timing-function: step-end;  
animation-timing-function: cubic-bezier(0.1, 0.7, 1.0, 0.1);  
animation-timing-function: steps(4, end);  
animation-timing-function: ease, step-start, cubic-bezier(0.1, 0.7, 1.0, 0.1);  
animation-timing-function: initial;  
animation-timing-function: inherit;
```



Свойство не наследуется.

animation-timing-function	
Значения:	
linear	Линейная функция, анимация происходит равномерно на протяжении всего времени, без колебаний в скорости.
функции Безье	
ease	Функция по умолчанию, анимация начинается медленно, разгоняется быстро и замедляется в конце. Соответствует <code>cubic-bezier(0.25,0.1,0.25,1)</code> .
ease-in	Анимация начинается медленно, а затем плавно ускоряется в конце. Соответствует <code>cubic-bezier(0.42,0,1,1)</code> .
ease-out	Анимация начинается быстро и плавно замедляется в конце. Соответствует <code>cubic-bezier(0,0,0.58,1)</code> .
ease-in-out	Анимация медленно начинается и медленно заканчивается. Соответствует <code>cubic-bezier(0.42,0,0.58,1)</code> .
cubic-bezier(x1, y1, x2, y2)	Позволяет вручную установить значения от 0 до 1. На этом сайте вы сможете построить любую траекторию скорости изменения анимации.

Повтор анимации: свойство `animation-iteration-count`

Свойство `animation-iteration-count` указывает, сколько раз проигрывается цикл анимации.

Начальное значение 1 означает, что анимация будет воспроизводиться от начала до конца один раз. Это свойство часто используется в сочетании со значением `alternate` свойства `animation-direction`, которое заставляет анимацию воспроизводиться в обратном порядке в альтернативных циклах.

`animation-iteration-count`

Значения:

`infinite`

Анимация проигрывается бесконечно.

число

Анимация будет повторяться указанное количество раз. Если число не является целым числом, анимация закончится в середине последнего цикла. Отрицательные числа недействительны. Значение `0` вызывает мгновенное срабатывание анимации.

```
animation-iteration-count: infinite;  
animation-iteration-count: 3;  
animation-iteration-count: 2.5;  
animation-iteration-count: 2, 0, infinite;
```

Направление анимации: свойство `animation-direction`

Свойство `animation-direction` определяет, должна ли анимация воспроизводиться в обратном порядке в некоторых или во всех циклах.

animation-direction	
Значения:	
normal	Все повторы анимации воспроизводятся так, как указано. Значение по умолчанию.
reverse	Все повторы анимации воспроизводятся в обратном направлении от того, как они были определены.
alternate	Каждый нечетный повтор цикла анимации воспроизводятся в нормальном направлении, каждый четный повтор воспроизводится в обратном направлении.
alternate-reverse	Каждый нечетный повтор цикла анимации воспроизводятся в обратном направлении, каждый четный повтор воспроизводится в нормальном направлении.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
animation-direction: normal;
animation-direction: reverse;
animation-direction: alternate;
animation-direction: alternate-reverse;
animation-direction: normal, reverse;
animation-direction: alternate, reverse, normal;
animation-direction: initial;
animation-direction: inherit;
```

Проигрывание анимации: свойство `animation-play-state`

Свойство `animation-play-state` определяет, будет ли анимация запущена или приостановлена.

- ✓ Остановка анимации внутри цикла возможна через использование этого свойства в скрипте JavaScript.
- ✓ Также можно останавливать анимацию при наведении курсора мыши на объект — состояние `:hover`.

```
animation-play-state: running;  
animation-play-state: paused;  
animation-play-state: paused, running, running;  
animation-play-state: initial;  
animation-play-state: inherit;
```

Задержка анимации: свойство `animation-delay`

Свойство `animation-delay` определяет, когда анимация начнется.

Задается в секундах s или миллисекундах ms.

```
animation-delay: 5s;  
animation-delay: 3s, 10ms;
```

Краткая запись анимации: свойство `animation`

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;`

Для воспроизведения анимации достаточно указать только два свойства — `animation-name` и `animation-duration`, остальные свойства примут значения по умолчанию.

Порядок перечисления свойств не имеет значения, единственное, время выполнения анимации `animation-duration` обязательно должно стоять перед задержкой `animation-delay`

Множественные анимации

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

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