Плавные переходы

**Длительность перехода, transition-duration**

Плавные переходы в CSS или transitions позволяют изменять значения свойств постепенно, «с нежностью».

Обычно эффект от изменения значений CSS-свойств виден мгновенно, но с помощью плавных переходов можно изменить это поведение и сделать процесс изменения значений достаточно длительным.

В отличие от [анимаций](https://htmlacademy.ru/courses/80), которые позволяют управлять любым количеством промежуточных состояний, с помощью transitions можно управлять только переходом между двумя состояниями: начальным и конечным.

Чтобы обозначить плавный переход в CSS, достаточно задать одно свойство: transition-duration — длительность перехода. Значения задаются в секундах (10s, 3s), долях секунды (0.1s, 0.03s) или миллисекундах (100ms, 333ms).

# Длительность перехода, шаг 2

Плавные переходы и анимацию можно применить только к некоторым CSS-свойствам. В основном это свойства, изменяющие размер, цвет, позицию элементов. В [статье на MDN](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_animated_properties) приведён перечень анимируемых свойств и характер их анимирования.

Если задана только длительность перехода transition-duration, то по умолчанию плавное изменение затрагивает все свойства элемента и для всех свойств происходит одновременно.

**transition-property: какие свойства изменять плавно?**

По умолчанию плавный переход действует на все анимируемые свойства элемента. Такое поведение соответствует значению all свойства transition-property, это значение по умолчанию.

Такое поведение часто нежелательно, особенно когда в правиле много свойств.

Можно указать, какие именно свойства нужно изменять плавно, перечислив их в transition-property через запятую:

transition-property: width; // плавно меняется только ширина

transition-property: width, height; // плавно меняются ширина и высота

При этом так же через запятую можно задавать переходам разных свойств разную длительность:

transition-property: width, height;

transition-duration: 1s, 5s; // ширина меняется за 1 секунду, высота за 5

Попробуем применить описанное свойство на практике. В этом примере у кнопки анимируются свойства background-color, transform и font-size.

# Задержка перехода, transition-delay

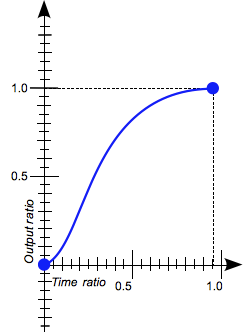
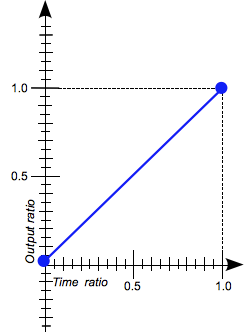
Ещё одно свойство переходов — transition-delay. Оно определяет задержку перед началом перехода и задаётся в секундах или миллисекундах, как и transition-duration.

# «Форма» перехода, transition-timing-function

Ещё одно свойство, влияющее на переход — transition-timing-function. Это свойство аналогично свойству animation-timing-function, которое разбирается в [части по анимациям](https://htmlacademy.ru/courses/80/run/23). Свойство transition-timing-function определяет с какой скоростью и ускорением будут меняться свойства во время перехода.

В предыдущих примерах переходы происходили с одинаковой динамикой. Мы меняли длительность и задержку перехода, но не «форму». Эта «форма» по умолчанию соответствует первому графику, из которого видно, что переход начинается медленно, затем ускоряется и к концу движения опять замедляется.

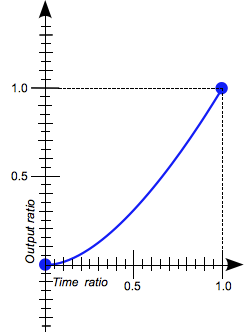
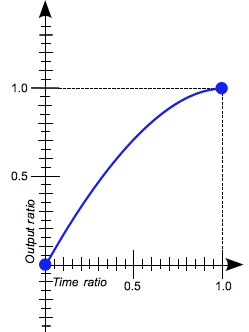
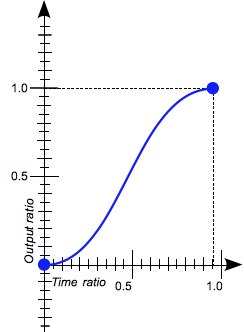
Так ведёт себя значение ease свойства transition-timing-function.

  
*ease*  
*linear*

Чтобы сделать переход равномерным, без ускорений и замедлений, используется значение linear. «Форма» линейного перехода приведена на втором графике.

# transition-timing-function, шаг 2

Вот ещё несколько форм переходов: ease-in, ease-out и ease-in-out.

  
*ease-in*  
*ease-out*  
*ease-in-out*

Из графиков видно, что при значении ease-in переход медленно начинается, а к концу ускоряется; при ease-out — начинается быстро, а к концу замедляется. Значение ease-in-out похоже на ease, то есть переход начинается и заканчивается медленно, но происходит это чуть-чуть интенсивнее.

# transition-timing-function, шаг 3

Названия linear, ease, ease-in и другие — это «псевдонимы» функций кубических [кривых Безье](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%B2%D0%B0%D1%8F_%D0%91%D0%B5%D0%B7%D1%8C%D0%B5):

cubic-bezier(0, 0, 1, 1) // это linear

cubic-bezier(0.42, 0, 1, 1) // это ease-in

В общем представлении cubic-bezier(x1, y1, x2, y2) значения x и y — это координаты точек кривых на графике. При этом верным считается значение x только в диапазоне от 0 до 1.

Существует [отличный сервис](http://cubic-bezier.com/), помогающий разобраться в функциональном представлении кривых Безье без необходимости штудировать учебники по математике.

А вот по этой [ссылке](http://easings.net/ru) можно найти целую коллекцию разных easing-функций на основе кривых Безье.

C помощью функции cubic-bezier мы можем задавать любые формы переходов.

# transition-timing-function, шаг 4

Ещё один возможный класс значений transition-timing-function — это steps.

Они позволяют задать «ступеньки», по которым будет идти переход. Синтаксис steps следующий:

transition-timing-function: steps(число\_шагов, направление);

Тут всё просто: число\_шагов — это целое число, за которое будет выполнен переход; направление может принимать значение start или end.

При заданном start первый шаг выполняется одновременно с началом перехода, а в случае c end последний шаг будет выполнен вместе с завершением перехода.

Кстати, переход можно описать в сокращённом виде свойством transition. Параметры перехода просто перечисляются через пробел: свойство, длительность, форма и задержка:

transition: width 1s ease-in 2s;

К примеру, переход тут применяется к ширине элемента, будет длиться 1 секунду с формой ease-in и задержкой перед началом в 2 секунды.