**CSS-анимация**

CSS3-анимацияпридаёт сайтам динамичность. Она оживляет веб-страницы, улучшая взаимодействие с пользователем. В отличие от CSS3-переходов, создание анимации базируется на ключевых кадрах, которые позволяют автоматически воспроизводить и повторять эффекты на протяжении заданного времени, а также останавливать анимацию внутри цикла.

**Правило keyframes и его синтаксис.**

Для того, чтобы указать для каких именно свойств элемента необходимо применять анимационные эффекты, используется отдельное правило @keyframes. Оно устанавливает ключевые кадры при анимации элемента, которые представляют собой конкретные стилевые свойства элемента в данный момент. Таким образом, анимация представляет собою не что иное, как переход от одного набора стилевых свойств элемента к другому.

**@keyframesexample\_1{**

**0% {background: red; height: 0px;}**

**50% {background: green; height: 400px;}**

**100% {background: blue; height: 200px;}**

**}**

**Имя анимации, css-свойство animation-name**

Сами анимационные свойства записываются в отдельном от @keyframes правиле на общих основаниях с другими правилами CSS. А для того, чтобы привязать анимационные свойства к конкретной анимации, т.е. к правилу @keyframes, используется свойство **animation-name**, принимающее в качестве значений либо список имен анимаций, которые должны быть применены к элементу, либо ключевое слово none, которое используется по умолчанию и отменяет анимацию. Имена в списке анимаций должны перечисляться через запятую. **animation-name: example\_1, example\_2;**

**Задержка анимации, css-свойство animation-delay**

В качестве значения свойство принимает время, указанное в секундах (s) или миллисекундах (ms). **animation-delay: 2s;**

**Длительность анимации, css-свойство animation-duration**

Чтобы установить продолжительность одного цикла анимации, необходимо использовать свойство **animation-duration**, которое также принимает в качестве значения время, указанное в секундах (s) или миллисекундах (ms) **animation-duration: 3s;**

**Повторение анимации, css-свойство animation-iteration-count**

Количество повторений анимации можно задать при помощи свойства animation-iteration-count, которое принимает в качестве значений либо положительное число, указывающее количество повторений, либо ключевое слово infinite, означающее воспроизведение анимации бесконечное число раз. При этом разрешается использовать не только целые значения, но и дробные. Например, если значение равно 1.5, то анимация совершит один полный цикл и потом завершится на половине второго цикла. По умолчанию анимация воспроизводится только один раз. animation-iteration-count: infinite;

**Направление анимации, css-свойство animation-direction**

Помимо количества повторений анимации можно задать и ее направление. Делается это при помощи свойства animation-direction, которое принимает значения в виде ключевых слов:

•**normal**–после завершения цикла, анимация сбрасывается в исходное состояние и стартует заново (используется по умолчанию);

•**alternate**–после завершения цикла, анимация начинает шаг за шагом воспроизводиться в обратном направлении;

•**reverse**–анимация начинается сразу с конца цикла, выполняя все шаги в обратном направлении, а затем сбрасывается опять в конец цикла;

•**alternate-reverse**–анимация начинается сразу с конца цикла, выполняя все шаги в обратном направлении, а затем начинает шаг за шагом воспроизводиться в прямом направлении, возвращаясь в конец цикла, который в данном случае является стартовой точкой .animation-direction: alternate;

**Плавность анимации,css-свойство animation-timing-function**

• **ease**–анимация ускоряется к середине, а затем замедляется к концу;

• **ease-in**–анимация начинает медленно ускоряться с самого начала и до конца;

• **ease-out**–анимация стартует ускоренно и замедляется к концу;

• **ease-in-out**–анимация медленно стартует и медленно заканчивается;

• **linear**–постоянная скорость на всем промежутке воспроизведения

• **cubic-bezier(n1,n2,n3,n4)** –поведение анимации будет зависеть от результата вычисления функции Безье, в качестве аргументов которой можно задавать четыре числа от нуля до единицы включительно;

**Постановка анимации на паузу,css-свойство animation-play-state**

Если в определенный момент нужно поставить анимацию на паузу, следует воспользоваться свойством animation-play-state, которое определяет два состояния анимации:

•**running**–анимация проигрывается;

• **paused**–анимация поставлена на паузу. Изменять значение свойства можно, например, при помощи JavaScript.

**css-свойство animation-fill-mode**

• **backwards**–во время старта анимации к элементу применяется стиль первого ключевого кадра, который будет сохраняться на весь период задержки анимации, установленный свойством animation-delay;

• **forward**–по окончании анимации к элементу применяется стиль последнего ключевого кадра;

**• both**–применяются оба значения: для момента запуска анимации применяется первое значение, т.е. backwards, а в момент финиша –второе, т.е. forward;

**• none–данное** значение применяется по умолчанию; стиль элемента до и после анимации остается неизменным, а если ранее было применено одно из значений backwardsили forward, то их действия будут отменены