**Цвет**

## С помощью RGB

Можно определить цвет, используя значения красной, зеленой и синей составляющей в десятичном исчислении. Каждая из трех компонент цвета принимает значение от 0 до 255. Также допустимо задавать цвет в процентном отношении, при этом 100% будет соответствовать числу 255. Вначале указывается ключевое слово rgb, а затем в скобках, через запятую указываются компоненты цвета, например rgb(255, 128, 128) или rgb(100%, 50%, 50%).

## RGBA

Формат RGBA похож по синтаксису на RGB, но включает в себя альфа-канал, задающий прозрачность элемента. Значение 0 соответствует полной прозрачности, 1 — непрозрачности, а промежуточное значение вроде 0.5 — полупрозрачности.

RGBA добавлен в CSS3, поэтому валидацию CSS-кода надо проводить именно по этой версии. Следует отметить, что стандарт CSS3 еще находится в разработке и некоторые возможности в нем могут поменяться. К примеру, цвет в формате RGB добавленный к свойству background-color проходит валидацию, а добавленный к свойству background уже нет. При этом браузеры вполне корректно понимают цвет для того и другого свойства.

**position в CSS**

Свойство position — это универсальное и мощное свойство. Оно позволяет установить или *изменить* положение элемента и содержит четыре возможных **значения**:

* static (значение по умолчанию)
* relative
* absolute
* fixed
* sticky

Часто используется наряду с четырьмя свойствами координат:

* left
* right
* top
* bottom

static

Это значение position **по умолчанию**: статичные элементы просто следуют естественному [потоку](https://webref.ru/course/positioning/flow). Любые значения left, right, top или bottom не оказывают влияния.

relative

Когда position установлено **относительно**, элемент может перемещаться относительно его **текущей позиции**.

**HTML**

<p>Едва мы спрятали лодку, Джа увлёк меня в джунгли, и вскоре мы

вышли на узкую, хорошо утоптанную тропу.</p>

<p class="second">Я ринулся вверх, пытаясь в последнем усилии

дотянуться до руки Джа, но ситик с такой силой рванул копьё на себя,

что оно вырвалось из рук мезопа, едва не сорвавшегося со скалы,

а я, всё ещё цепляясь за древко, полетел вниз, прямо в

раскрытую пасть зверюги.</p>

<p>Он уже перешёл на бег, замахиваясь на ходу тяжёлым копьём.</p>

**CSS**

p { border: 1px solid blue; }

Давайте переместим второй абзац:

.second { position: relative; border-color: red; left: 20px; top: 10px; }

Красный абзац переместился на 20px слева **направо** и на 10px сверху **вниз** относительно его *естественного* положения, где он *полагался* быть.

Обратите внимание, что синие абзацы не сдвинулись вообще. При использовании относительного позиционирования, красный абзац может свободно перемещаться, не нарушая макет. Единственное, что сдвигается со своего места, так это сам элемент. Все остальные элементы **не знают, что элемент переместился**.

absolute

Когда position устанавливается **абсолютно**, элемент может перемещаться относительно **первого позиционированного предка**.

Позиционированного? Что такое позиционированный элемент?

**Позиционированный** элемент — это тот, у которого значение position установлено как relative, absolute или fixed. Таким образом, если положение не задано статичным, то элемент **позиционированный**.

Характеристикой *позиционированного* элемента является то, что он может выступать в качестве **точки отсчёта для своих дочерних элементов**.

Давайте представим простую иерархию:

**HTML**

<section>

Я позиционирован относительно.

<p>

Я позиционирован абсолютно!

</p>

</section>

**CSS**

section {

background: gold;

height: 200px;

padding: 10px;

position: relative; /\* Превращает <section> в точку отсчёта для <p> \*/

}

p {

background: limegreen;

color: white;

padding: 10px;

position: absolute; /\* Делает <p> свободно перемещаемым \*/

bottom: 10px; /\* 10px снизу \*/

left: 20px; /\* 20px слева \*/

}



Жёлтый раздел имеет высоту 200px, а позиционирование задано как relative, что превращает его в **точку отсчёта для всех дочерних элементов**.

Зелёный абзац позиционирован как absolute и может свободно перемещаться *относительно* жёлтого раздела. Установив оба значения left и bottom, мы *переместим* его в левый нижний угол.

Что произойдёт, если мы одновременно установим left и right?

Если ширина не задана, применение left: 0 и right: 0 **растягивает элемент на всю ширину**. Это эквивалент установки left: 0 и width: 100%.

Если ширина установлена, то значение right отбрасывается.

fixed

Когда position задано как fixed, оно действует как **абсолютное**: вы можете установить координаты слева/справа и сверху/снизу.

Единственное отличие состоит в том, что **точкой отсчёта является окно просмотра**. Это означает, что фиксированный элемент *не перемещается* со страницей, он *фиксируется* на экране.

sticky

Когда position задано как sticky, оно действует как sticky: вы можете установить координаты слева/справа и сверху/снизу.

Единственное отличие состоит в том, что **точкой отсчёта является родительский контейнер с позиционированием**. Это означает, что липкий элемент *не перемещается вместе со всем* с окном браузера, он *фиксируется* внутри родителя.

z-index

Любые позиционированные элементы на веб-странице могут накладываться друг на друга в определенном порядке, имитируя тем самым третье измерение, перпендикулярное экрану. Каждый элемент может находиться как ниже, так и выше других объектов веб-страницы, их размещением по z-оси и управляет z-index. Это свойство работает только для элементов, у которых значение position задано как absolute, fixed или relative.

В качестве значения используются целые числа (положительные, отрицательные и ноль). Чем больше значение, тем выше находится элемент по сравнению с теми элементами, у которых оно меньше. При равном значении z-index, на переднем плане находится тот элемент, который в коде HTML описан ниже. Хотя спецификация и разрешает использовать отрицательные значения z-index, но такие элементы не отображаются в браузере Firefox до версии 2.0 включительно.

Кроме числовых значений применяется auto — порядок элементов в этом случае строится автоматически, исходя из их положения в коде HTML и принадлежности к родителю, поскольку дочерние элементы имеют тот же номер, что их родительский элемент. Значение inherit указывает, что оно наследуется у родителя.