Все о frontend.

Разделы страницы.

Тег main содержит основную информацию страницы. Притом этот тег можно использовать один раз.

Тег aside содержит косвенно-связанную информацию от main, к примеру, ссылки на статьи, сноски, рекламу или боковая панель.

Тег article – содержит уникальные элементы страницы и обычно располагается в main. К примеру, используется для статей.

Тег footer – должен содержит подвал страницы и обычно содержит адреса. Ограничение на использование внутри footer.

Тег nav – должен содержать основные навигационные ссылки. Данный тег разрешает множественное использование, но всегда следует выделять главный nav.

Тег header – должен содержать шапку страницы. Внутри себя он не должен использовать header, address, footer и main.

Тег section – должен логически группировать элементы, когда не подходят выше описанные варианты. Используется крайне редко.

Тег div служит для стилевого обозначения блока, в то время span служит для стилевого обозначения строчного элемента.

Основы CSS

Наследование стилей

body {

font-size: 12px;

color: grey;

}

body span {

font-size: 2em;

}

Из этого примера можно увидеть, наследование стилей от своего предка, но наследуются только некоторые свойства:

* color
* font
* line-height

Некоторые теги (к примеру, <a>) и вовсе не наследуют стили

Единицы размера

* px позволяет указать размер в пикселях
* % позволяет указать размер в процентах относительно текущего положения
* em позволяет указать размер относительно предка. К примеру, выше мы указываем, что стиль должен быть в два раза больше
* ret тоже что и em, но наследуется от корня (html)

Приоритет селекторов

1. Стиль в атрибуте style
2. Идентификаторы
3. Классы
4. Теговые

Расширение селекторов

Для того чтоб переиспользовать или задать дополнительные стили используются *расширения селекторов*. Одним из расширений является наследование, когда у одного элемента дается стиль для него подэлемента:

html a { }

a.header {}

.foo #bar {}

Но также можно задать стиль сразу нескольким элементам:

.foo, #bar {}

Блочная модель.

Свойство display позволяет менять тип элемента:

* block – ставит тип, как блочный
* inline - ставит тип, как строчный
* inline-block – это смесь блочного и строчного типа. У таких элементов можно ставить padding, margin, border, но притом они не будут занимать все возможное пространство
* none – позволяет полностью убрать элемент

Для display: none есть альтернатива в виде visibility:

* visible – делает элемент видимым
* hidden - делает элемент прозрачным, но его пространство все равно занято

Размер элемента можно определить свойствами, указав оба элемент, станет полностью статичным:

1. height
2. width

Для того чтоб объект не выходил за рамки используется overflow:

* hidden – скрывает, все что выходит за предел
* scroll – подставляет ползунки
* visible – показывает все, что выходит за предел
* auto – сам подставит при необходимости scroll

Свойство overflow и указание фиксированных размеров – это плохая практика, которую надо минимизировать. Альтернатива использовать min-width, max-width и max-height, min-height

Границы

Свойство border представляет собой границу для всех сторон. Также есть дополнительные типа border-left, которые позволяют задать границу для одной стороны

* border-color: задает цвет
* border-size : задает размер
* border-radius : задаем радиус
* border-style : стиль границы (сплошной – solid, double – двойной, dashed – пунктир и т.д)

Внутренние отступы

Свойство padding позволяет установить внутренние отступы внутри элемента, т.е. насколько далеко должна быть граница от содержимого, для задания отдельных сторон есть padding-left и т.д.

padding: *значение для всех сторон*

padding: *верх низ*, *лево право*

Внешние отступы

Свойств margin позволяет задать расстояние между объектами. У него соответственно есть margin-left и другие. Если располагаем два элемента по bottom: 30px и top: 15px, то мы получим разницу между элементами в 45px.

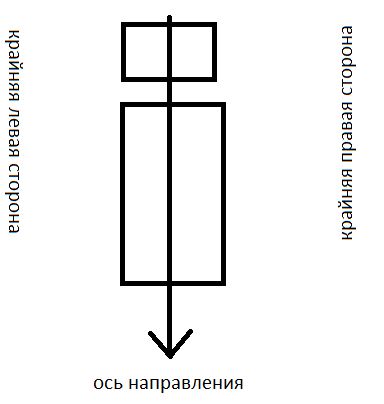
Свойство box-sizing

По умолчанию расчет ширины элемента идет по сумме width, border, margin, padding и для высоты соответственно. Если же способ подсчета нужно нарушить, то можно применить box-sizing:

* content-box: не считает значения margin, border, padding
* border-box: не считает значение margin, поэтому и называется border, т.к. на border все заканчивается

Позиционирование

По умолчанию объекты располагаются согласно *потоку*, т.е. каждый элемент накладывается друг друга. Элементы располагаются по *оси направления*, все они выравниваются вертикально.

Свойство float

Свойство float позволяет изменить поток, прижав элемент к левой стороне или правой соответственно.

Но float по сути своей изменяет все окружающие элементы. Для того чтоб запретить все изменения есть свойство clear, дружественное float:

* left – запрещает выравнивать элемент по левой границе
* right – запрещает выравнивать элемент по правой границе
* both – запрещает выравнивать по обоим бортам

Основным применением в верстке – это создание двухколоночных макетах. В остальном это свойство пользуется средней популярностью, так как он непредвиден, и поддерживать макет, основанный на float трудно.

Свойство position.

Свойства position позволяет установить тип переноса элемента, у него есть следующие значения:

* static - значение по умолчанию, которое придерживает элемент к стандартному потоку.
* relative
* fixed
* absolute

Для того чтоб переместить элемент имеются свойство top, bottom, left и right.

Относительное позиционирование

Значение relative – позволяет изменить положение элемента от своей изначальной фиксированной точки, также он резервирует свое изначальное место и на нем остается пустота.

К примеру, мы вытаскиваем 2 кирпич из стопки 5 кирпичей, то 1 кирпич не подтянется к 3 кирпичу. Этим и отличаются fixed и absolute, тем, что они подтягивают элементы.

Также relative позволяет безгранично перетаскивать элемент, а если у его родителя стоит overflow: hidden, то при выходе из границ элемент будет отрезаться.

В системе координат при относительном позиционировании, если двигать элемент по top, то элемент наоборот упадет и т.д.

Абсолютное позиционирование

Свойство absolute, работает без фиксированной точки, и передвижение осуществляется по абсолютным координатам. Это значение редко используется, ведь она будет хорошо работать, только на мониторе верстальщика.

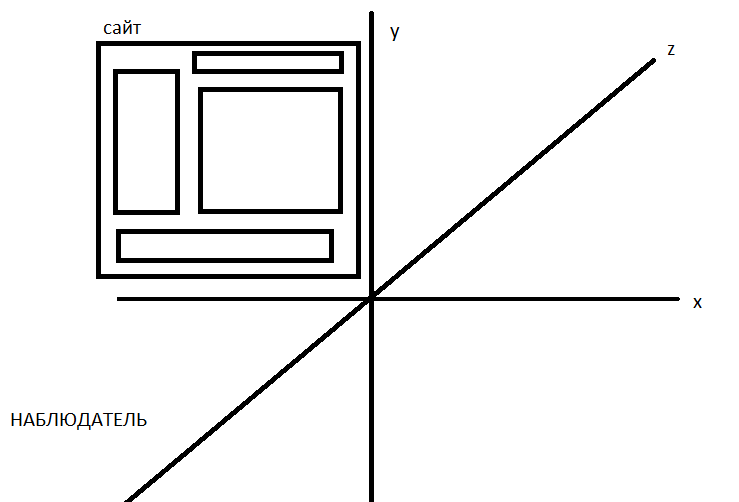
Однако в связке с relative, ее можно использовать. Если нам поставить на крайний правый конец родительского поля, то можно задать top и left – 0.

Связку relative-absolute можно описать, как позиционирование с фиксированной точкой, но возможностью из него выйти

Фиксированное позиционирование

Значение fixed – по поведению похож на absolute, но он будет всегда находиться на сайте, потому что он не подчиняется scroll.

Он часто применяет, когда нужно сделать баннер с рекламой, окно дополнительной связи, показать новое сообщение и т.д.

Свойство z-index

Пространство сайта можно изобразить как координаты с дополнительной осью z. На одном краю сидит пользователь, а на противоположные элементы.

Более близкие элементы по оси z, соответственно будут покрывать сзади идущие и наоборот.

Для работы с осью z имеется свойство z-index, которое будет работать, только не с static.

z-index: значение по оси z

Псевдоклассы и псевдоэлементы

Псевдоэлементы before и after

Эти псевдоклассы позволяют добавить *псевдоэлемент* после элемента и перед элементом соответственно, но главное то, что псевдоэлементы будут в пределах рамок. Вместе с after и before используют свойство *content*, позволяющий вставить контент.

* строчка
* url(*путь*): *путь* на какой-либо элемент
* open-quote: вставляет открывающую скобку
* close-quote: вставляет закрывающую скобку
* no-open-quote: убирает открывающую скобку
* no-close-quote: убирает закрывающую скобку
* attr (*атрибут*): позволяет ставить значение какого-либо *атрибута*.

К примеру используем after и before для того чтоб оформить цитату по краям скобками:

blockquote::before,

blockquote::after {

content: "“";

font-size: 40px;

color: #c1172c;

line-height: 0px;

}

blockquote::after {

content: "”";

}

Структурные псевдоклассы

|  |  |
| --- | --- |
| first-child | применяется только для первого элемента |
| last-child | применяется только для последнего элемента |
| nth-child(*номер*) | применяется только для элемента с *номером* |
| nth-child(odd) | применяется только для нечетных элементов |
| nth-child(even) | применяется только для четных элементов |
| nth-child(*число*n) | применяется для элементов четных *числу*. После *числа* должен стоять n |
| nth-child(*число*n + 1) | применяется для элементов четных произведению сумме |

Остальные избранные псевдоклассы.

Псевдокласс focuse позволяет изменить стиль при фокусировании элемента.

Псевдокласс visited позволяет задать стиль для посещенных ссылок

Псевдокласс link позволяет задать стиль для не посещенных ссылок

Псевдокласс disabled применяется для элементов с логическим атрибутом disabled, т.е. заблокированным

Стилизация отдельных элементов

В это разделе, речь идет о стилизации конкретных типов элементов, а также об использовании градиентов, фона и теней.

Стилизация таблиц

Для того чтоб указать сдваивалась ли граница ячеек используется свойство border-collapse:

* collapse – только одна граница между ячейками
* separator – на границе ячеек образуется две границы.

Для того чтоб задать ширину можно использовать свойство width, а height нельзя задать, т.к. высота считается в зависимости значений. Альтернатива – это использование padding. Если border-collapse separator, то можно указать дистанцию между границами ячеек свойством border-spacing.

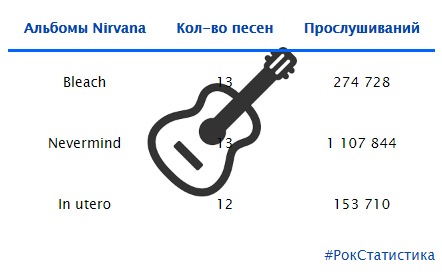
Для того чтоб скрыть пустые ячейки используется свойство empty-cells:

* hide – не отображает пустой элемент
* show – отображает пустой элемент

Для компоновки таблицы используется table-layout:

* auto – значение по умолчанию, которое считает ширину вместе с содержимым и значением.
* fixed – фиксирует ширину для всех ячеек исходя из переданного width

Примеры использования:

Стилизация списков

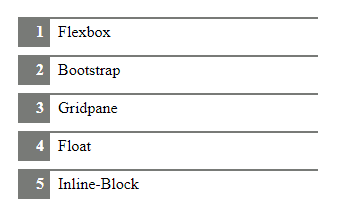
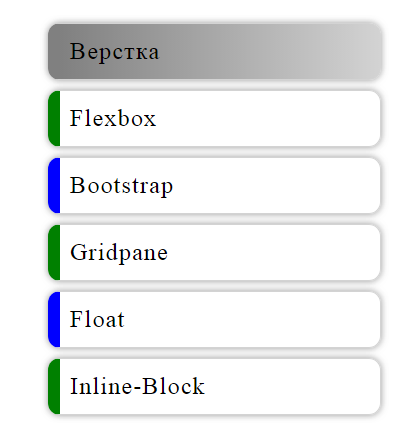
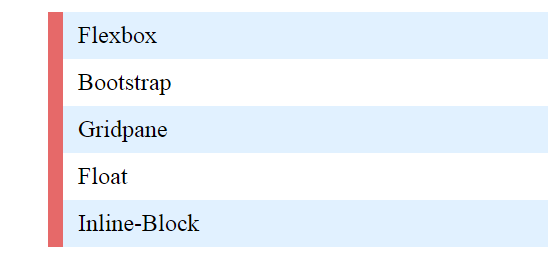
Для того чтоб придать стиль маркеру в списке используется свойство list-style-type. Вот лишь некоторые значения:

* disc – закрашенный круг
* armenian – армянская нумерация
* circle – не закрашенный круг
* decimal – цифры
* katakana – японская нумерация
* lower-alpha: a, b, c, d, e
* upper-alpha: A, B, C, D, E
* square – квадрат
* none – отключает нумерацию

Вместо маркера можно поставить какую-либо картинку или градиентом свойством list-style-image.

Чтоб расположить маркер используется свойство list-style-position:

* inside – вставляет маркер в текст
* outside – вставляет маркер слева от текста



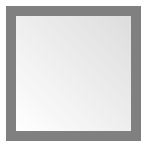
Стилизация текста

|  |  |  |
| --- | --- | --- |
| font-family: имя шрифта [, имя шрифта[, ...]] | позволяет задать стиль для строк | следует вставлять несколько шрифтов |
| line-height | позволяет задать размер для линии строки | кроме стандартных обозначений, можно передать просто число |
| font-weight: normal | bold | позволяет сделать строку жирной | normal делает строку чистой |
| font-style: normal | italic | позволяет сделать строку курсивной |
| text-decoration: none, overline, underline, line-through | позволяет наложить декорацию на строку | none позволяет сделать строку чистой |
| text-indent | позволяет указать отступ | |
| text-align: center, left, right | позволяет сгруппировать строку | |
| text-overflow: clip, ellipse, *строчка* | поведение при переполнении текста | clip – скрывает значение  ellipse – обрезает значение и ставить троеточие  *строчка* – выводится для показа обрезанного текста |
| direction: ltr, rtl | изменяется направление | ltr значение по умолчанию. |

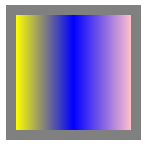
Градиенты

Для построения линейных градиентов используется функция linear-gradient(). С её аргументами:

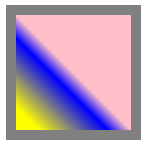
1. Размер угла или направление в словах (значения по умолчанию 180deg)
2. Цвета и дополнительно процент целостности цвета



background: linear-gradient (to left bottom, #dddddd, #ffffff)



background: linear-gradient (90deg, yellow, blue, pink)



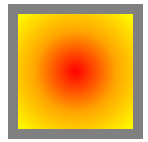
background: linear-gradient (to top right, yellow 10%, blue 40%, pink 50%)



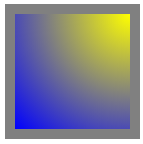
background: linear-gradient (orange, yellow)

Для создания радиальных градиентов используется функция radial-gradient(), похожая на линейный аналог. В линейных градиентах имеется только одна ось, в то время, когда в радиальных градиентах множество осей, направленных в разные направления.

Также в радиальных градиентах имеется *начальная точка*, которой пойдут лучи и *конечная точка*, в которую попадают лучи.

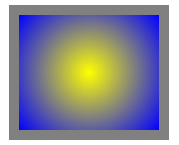
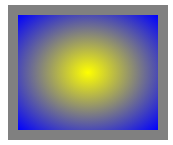


background: radial-gradient(red, orange, yellow);



background: radial-gradient (at top right, yellow, blue);

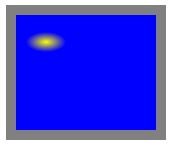
Иллюстрация разницы между ellipse и circle соответственно:



background: radial-gradient (ellipse, yellow, blue);

background: radial-gradient (circle, yellow, blue);

Также есть возможность указывать *фиксированные* значения:



background: radial-gradient (20px 10px at 30px 27px, yellow, blue);

Фон

Свойство background-image позволяет фоном какую-либо картинку. Его значения:

* url – путь для файла или файла в интернете
* linear-gradient()
* repeating-radial-gradient()

Для того повторение фона, имеется свойство background-repeat:

* no-repeat: не повторяет фон
* repeat-x: повторяет фон по ширине
* repeat-y: повторяет фон по высоте
* repeat: повторит фон, сколько возможно

Фон также можно закрепить свойством background-attachment:

* scroll, такой фон, можно прокручить вместе с элементом, значение по умолчанию
* fixed, такой фон, нельзя прокручить

Для позиционирования фона используется свойство background-position. У него множество понятных значений типа center, который располагает элемент в цент, left, располагающий фон в левый край, а также значения проценты, px и cm.

По умолчанию фон рисуется под рамками элемента, включая рамку. Для указания типа прорисования элемента используется свойство background-clip:

* border-box: значение по умолчанию
* padding-box: фон будет в пределах внутренних отступов
* content-box: фон будет в пределах контента

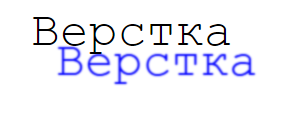
Для того чтоб задать размер фона используется свойство background-size.

* %, px, cm
* auto – определит размер автоматически
* cover – масштабирует фон, для того чтоб он занял весь экран, сохраняя пропорция, но при этом обрезая фон
* сontaint – масштабирует фон с сохранением пропорций.

Тени и прозрачность.

Свойство opacity позволяет указать прозрачность. Для нее значения варьируются полностью видимого 1 и не видимого 0.

Свойство text-shadow позволяет создать тень для текста.

1. *x* – значение смещения по ширине
2. *y* – значение смещения по высоте
3. *размытость* насколько размыть тень, для обозначения размытости используется px, все что выше 1, то тень размывается. Необязательный параметр.
4. *цвет* – цвет для тени
5. 

text-shadow: 25px 30px 2px blue;

Анимация

Переходы

Переход – это анимация, которая позволяет управлять переходом от одного стиля к другому, делая его более плавным. Единственным необходимым свойством является transition-duration, которому необходимо передать время в секундах или миллисекундах.

Возможно поставить паузу перед переходом свойством transition-delay, которому опять же нужно передать время в секундах или миллисекундах.

Не все свойства способны к переходу ([список способных к переходу](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_animated_properties)) и для того чтоб управлять тем, что должно переходить есть свойство transition-property, встроенные значения:

* all – использует все возможные свойства
* none – отключает переход всех свойств

Свойство transition-timing-function позволяет управлять скорость перехода:

* linear – реализует постоянную скорость
* **ease – сначала медленно, потом быстро и потом медленно**
* ease-in-out – аналог ease с плавными переходами
* ease-out – быстро разгоняется и медленно заканчивает
* ease-in – противоположность ease-out