

CSS

Box-model

Box-sizing

Media-query

Position

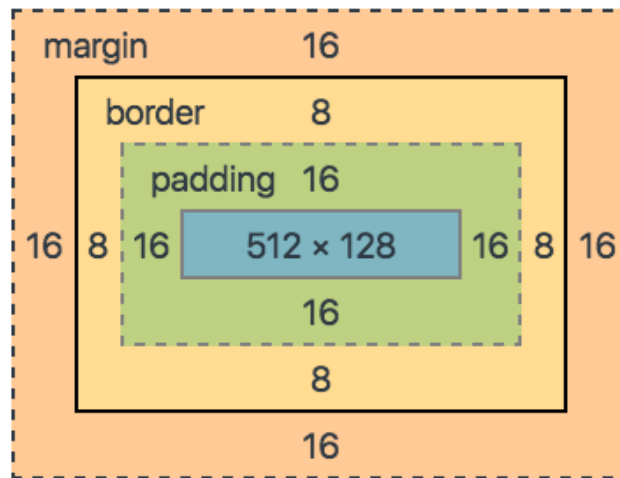
Adaptivity

Cross-browser

Git

Box-model

В HTML-документе каждому элементу на странице соответствует прямоугольная область (бокс или блок). Движок рендеринга в браузере определяет размеры и положение боксов на странице, а также их свойства вроде цвета, фоновой картинки для того, чтобы отобразить их на экране.



В языке CSS есть специальная боксовая модель (также блоковая модель или блочная модель, англ. box model), которая описывает, из чего состоит бокс и какие свойства влияют на его размеры.

В ней у каждого бокса есть 4 области:

margin (внешние отступы),

border (рамка),

padding (внутренние поля),

content (контент или содержимое).

Внутренняя область элемента (**content area**) содержит текст и другие элементы, расположенные внутри (*контент*).

У неё часто бывает фон, цвет или изображение (в таком порядке: **фоновый цвет скрывается под непрозрачным изображением**), и она находится внутри *content*;

её размеры называются *ширина контента* (*content width* или *content-box width*), и *высота контента* (*content height* или *content-box height*). Иногда еще говорят «внутренняя ширина/высота элемента»

По умолчанию, если CSS-свойство [box-sizing](#) не задано, размер внутренней области с содержимым задается свойствами [width](#), [min-width](#), [max-width](#), [height](#), [min-height](#) and [max-height](#). Если же свойство [box-sizing](#) задано, то оно определяет, для какой области указаны размеры.

Поля элемента (**padding area**) — это пустая область, окружающая контент. Она может быть залита каким-то цветом, покрыта фоновой картинкой, а её границы называются края полей (*padding edge*).

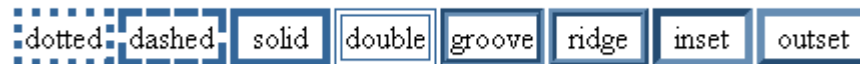
Размеры полей задаются по отдельности с разных сторон свойствами [padding-top](#), [padding-right](#), [padding-bottom](#), [padding-left](#) или общим свойством [padding](#).

Область рамки (**border area**) окружает поля элемента, а ее граница называется края рамки (*border edge*). Ширина рамки задается отдельным свойством [border-width](#) или в составе свойства [border](#). Размеры элемента с учетом полей и рамки иногда называют внешней шириной/высотой элемента.

border: [[border-width](#) || [border-style](#) || [border-color](#)] | inherit

border-top, border-bottom, border-left, border-right.

Для управления видом border предоставляется несколько значений border-style



Отступы (**margin area**) добавляют пустое (**прозрачное**) пространство вокруг элемента и определяют расстояние до соседних элементов.

Величина отступов задается по отдельности в разных направлениях свойствами [margin-top](#), [margin-right](#), [margin-bottom](#), [margin-left](#) или общим свойством [margin](#).

Отступы двух соседних элементов, расположенных друг над другом или вложенных друг в друга, могут накладываться. Это называется схлопывание границ ([margin collapsing](#)).

Схлопываются только вертикальные отступы.

<https://codepen.io/anon/pen/NVVKze>

codepen.io

outline

<https://codepen.io/anon/pen/BMExXZ>

В отличие от border, рамка outline не участвует в блочной модели CSS. Она не занимает места и не меняет размер элемента. Поэтому его используют, когда хотят добавить рамку без изменения других CSS-параметров.

Также, в отличие от border, рамку outline можно задать только со всех сторон: свойств outline-top, outline-left не существует.

outline: outline-color || outline-style || outline-width | inherit

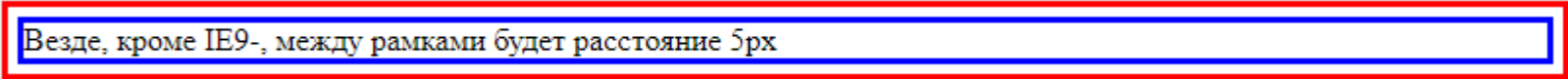
outline-color Задает цвет линии в любом допустимом для CSS формате.

outline-style Стиль линии.

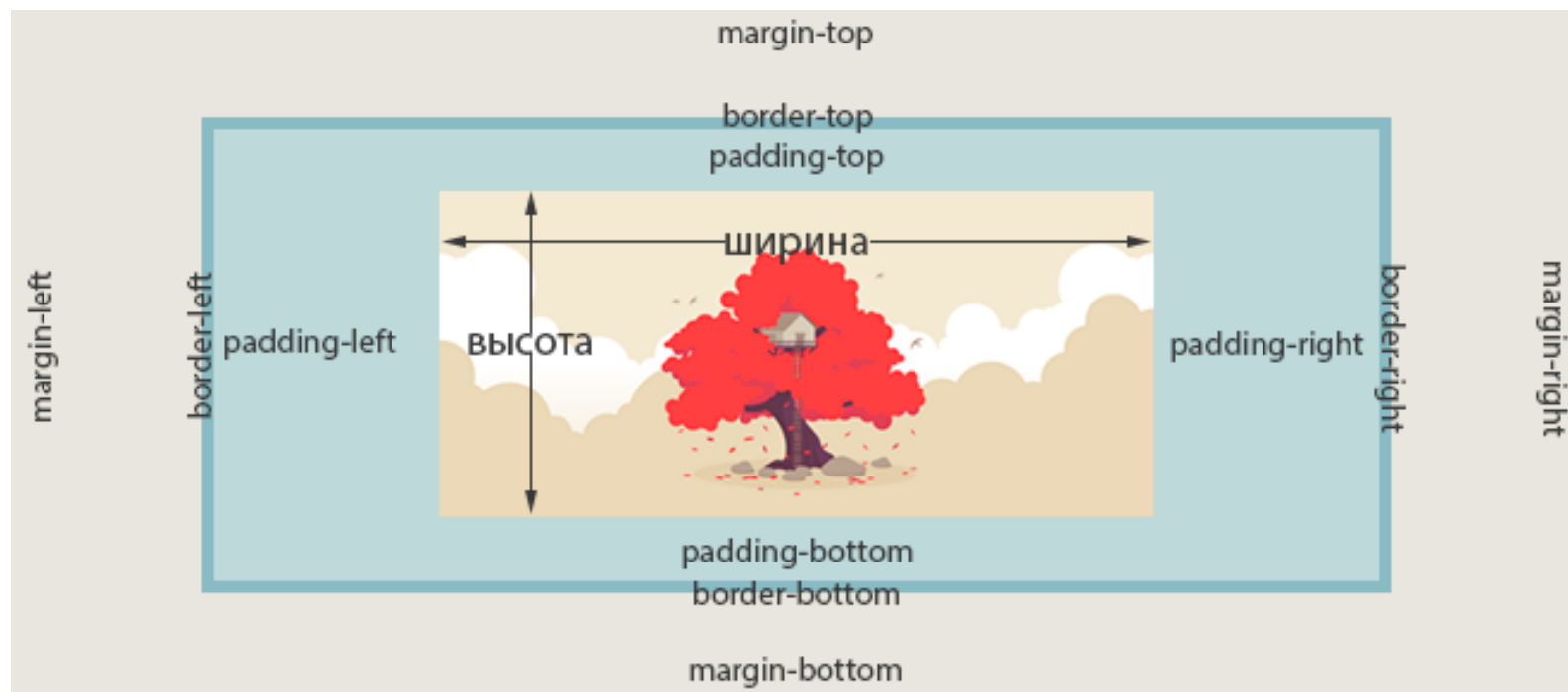
outline-width Толщина границы.

Inherit Наследует значение родителя.

Все браузеры, кроме IE9-, также поддерживают свойство outline-offset, задающее отступ outline от внешней границы элемента:



Везде, кроме IE9-, между рамками будет расстояние 5px



box-sizing

Применяется для изменения алгоритма расчета ширины и высоты элемента.

Согласно спецификации CSS ширина блока складывается из ширины контента (width), значений отступов (margin), полей (padding) и границ (border). Аналогично обстоит и с высотой блока.

Свойство box-sizing позволяет изменить этот алгоритм, чтобы свойства width и height задавали размеры не контента, а размеры блока.

box-sizing: content-box | border-box | padding-box | inherit

content-box

Основывается на стандартах CSS, при этом свойства width и height задают ширину и высоту контента и не включают в себя значения отступов, полей и границ.

border-box

Свойства width и height включают в себя значения полей и границ, но не отступов (margin). Эта модель используется браузером Internet Explorer в режиме несовместимости.

padding-box

Свойства width и height включают в себя значения полей, но не отступов (margin) и границ (border).

inherit

Наследует значение родителя.

<https://codepen.io/anon/pen/XOwMra>

<https://codepen.io/maattt/full/oXpjzv>

https://www.w3schools.com/css/css_boxmodel.asp

https://www.w3schools.com/css/css_outline.asp

https://www.w3schools.com/css/css_padding.asp

https://www.w3schools.com/css/css_margin.asp

https://www.w3schools.com/css/css_border.asp

CSS Media Queries

Текущий синтаксис HTML5 и CSS3 напрямую ссылается на первую спецификацию Media Queries.

Медиа-запрос состоит из ключевого слова, описывающего тип устройства (необязательный параметр) и выражения, проверяющего характеристики данного устройства.

Из всех характеристик чаще всего проверяется ширина устройства `width`. Медиа-запрос является логическим выражением, которое возвращает **истину или ложь**.

Медиа-запросы могут быть добавлены следующими способами:

С помощью HTML:

```
<link rel="stylesheet" media="screen and (color)" href="example.css">
```

С помощью правила `@import` внутри элемента `<style>` или внешней таблицы стилей:

```
@import url(color.css) screen and (color);
```

В коде страницы:

```
<style>  
  @media (max-width: 600px) {  
    .sidebar {display: none;}  
  }  
</style>
```

Внутри таблицы стилей style.css:

```
@media (max-width: 600px) {  
  .sidebar {display: none;}  
}
```


Логические операторы

С помощью логических операторов можно создавать комбинированные медиазапросы, в которых будет проверяться соответствие нескольким условиям.

Оператор and

```
@media (min-width: 600px) and (max-width: 800px) { /* CSS-стили */; }
```

```
@media screen and (max-width: 600px) { /* CSS-стили */; }
```

```
@media all(устройств) and (max-width: 600px) {...}
```

Оператор запятая

Оператор запятая работает по аналогии с логическим оператором or

```
@media screen, projection { /* CSS-стили */; }
```

В данном случае CSS-стили, заключенные в фигурные скобки, сработают только для экранных или проекционных устройств.

Оператор not

Логическое НЕ. Указывается для отрицания условия.

Стиль для всех устройств кроме смартфонов

@media all and (not handheld) { ... }

Оператор not имеет низкий приоритет и оценивается в запросе последним, поэтому выражение

@media not all and (color) { ... }

следует понимать как

@media not (all and (color)) { ... }

а не

@media (not all) and (color) { ... }

Тип носителя

All Подходит для всех типов устройств.

Print Предназначен для страничных материалов и документов, просматриваемых на экране в режиме предварительного просмотра печати.

Screen Предназначен в первую очередь для экранов цветных компьютерных мониторов.

Speech Предназначен для синтезаторов речи.

Характеристики носителя

Width Проверяет ширину области просмотра. Значения задаются в единицах длины, px, em и т.д., например, (width: 800px). Обычно для проверки используются минимальные и максимальные значения ширины. min-width применяет правило если ширина области просмотра больше значения, указанного в запросе, max-width — ширина области просмотра меньше значения, указанного в запросе.

Height Проверяет высоту области просмотра. Значения задаются в единицах длины, px, em и т.д., например, (height: 500px). Обычно для проверки используются минимальные и максимальные значения высоты. min-height применяет правило если высота области просмотра больше значения, указанного в запросе, max-height — высота области просмотра которого меньше значения, указанного в запросе.

aspect-ratio Проверяет соотношение ширины к высоте области просмотра. Широкоэкранный дисплей с соотношением сторон 16:9 может быть помечен как (aspect-ratio: 16/9). min-aspect-ratio проверяет минимальное соотношение, max-aspect-ratio — максимальное соотношение ширины к высоте области просмотра.

Orientation Проверяет ориентацию области просмотра. Принимает два значения: (orientation: portrait) и (orientation: landscape).

resolution Проверяет разрешение экрана (количество пикселей). Значения также могут проверять количество точек на дюйм (dpi) или количество точек на сантиметр (dpcm), например, (resolution: 300dpi).

min-resolution проверяет минимальное разрешение экрана, max-resolution — максимальное.

Color Проверяет количество бит на каждый из цветовых компонентов устройства вывода. Например, (min-color: 4) означает, что экран конкретного устройства должен иметь 4-битную глубину цвета. min-color проверяет минимальное количество бит, max-color — максимальное количество бит.

color-index Проверяет количество записей в таблице подстановки цветов. В качестве значения указывается положительное число, например, (color-index: 256). min-color-index проверяет минимальное количество записей, max-color-index — максимальное количество записей.

Monochrome Проверяет количество битов на пиксель монохромного устройства. Значение задается целым положительным числом, например, (min-monochrome: 8). min-monochrome проверяет минимальное количество битов, max-monochrome — максимальное количество битов.

Метатег viewport

Для управления разметкой в мобильных браузерах используется метатег viewport. С помощью метатега viewport можно контролировать размер окна просмотра и масштаб.

Страницы, адаптированные для просмотра на разных типах устройств, должны содержать в разделе <head> метатег viewport.

<meta name="viewport" content="width=device-width, initial-scale=1.0">

Свойство width определяет виртуальную ширину окна просмотра, значение device-width — физическую ширину устройства. Другими словами, width отражает значение document.documentElement.clientWidth, а device-width — screen.width.

При первой загрузке страницы свойство `initial-scale` управляет начальным уровнем масштабирования, `initial-scale=1` означает, что 1 пиксель окна просмотра = 1 пиксель CSS.



На какие размеры экрана нужно ориентироваться

<https://responsivedesign.is/develop/browser-feature-support/media-queries-for-common-device-breakpoints/>

Стратегии использования медиа-запросов

Для создания дизайна, позволяющего лучшим образом отображать сайт на различных устройствах, используют общие стратегии медиа-запросов:

- 1) Уменьшение количества колонок (столбцов) и постепенная отмена обтекания для мобильных устройств.
- 2) Использование свойства `max-width` вместо `width` при задании ширины блока-контейнера.
- 3) Уменьшение полей и отступов на мобильных устройствах (например, нижних отступов между заголовком и текстом, левого отступа для списков и т.п.).
- 4) Уменьшение размеров шрифтов для мобильных устройств.
- 5) Превращение линейных навигационных меню в раскрывающиеся.
- 6) Скрытие второстепенного содержимого на мобильных устройствах с помощью `display: none`.
- 7) Подключение фоновых изображений уменьшенных размеров (или отключение фоновых изображений).

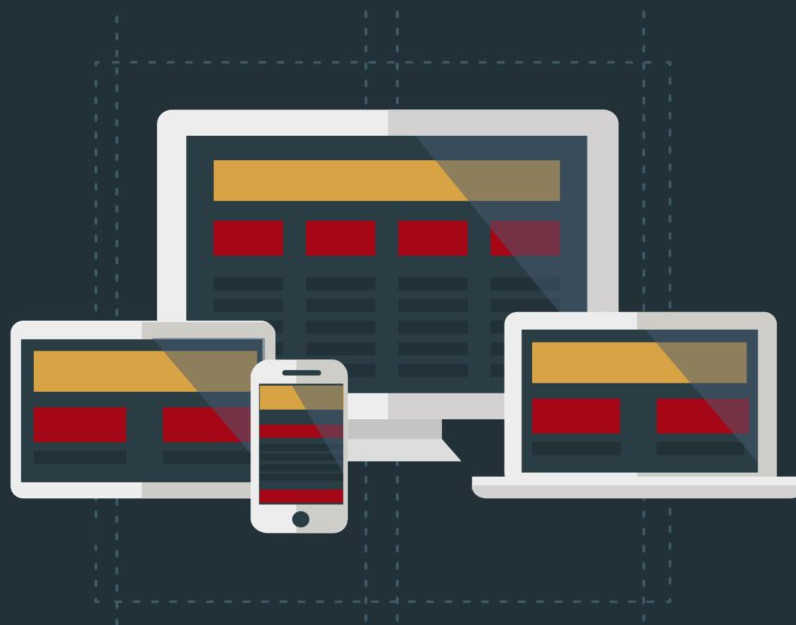
Адаптивность

Адаптивная верстка меняет дизайн страницы в зависимости от поведения пользователя, платформы, размера экрана и ориентации девайса и является неотъемлемой частью современной веб-разработки.

Она позволяет существенно экономить и не отрисовывать новый дизайн для каждого разрешения, а менять размеры и расположение отдельных элементов.

Проверка адаптивности

https://search.google.com/test/mobile-friendly?utm_source=mft&utm_medium=redirect&utm_campaign=mft-redirect



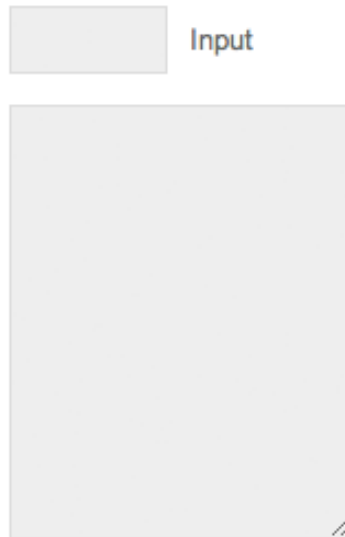
Простые техники адаптивной верстки

Максимальная и минимальная ширина

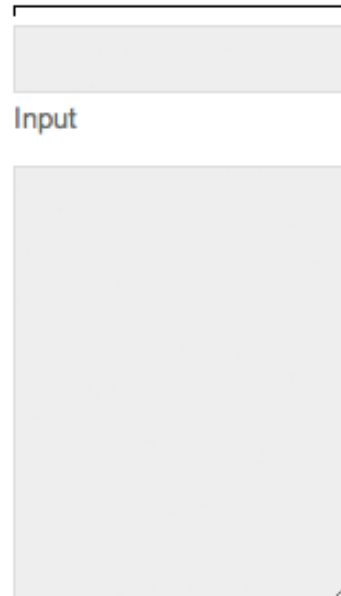
Max-width помогает определить максимально возможную ширину объекта.

Min-width — противоположность max-width, позволяет задать минимальную ширину объекта.

without min-width



min-width



Относительные значения

Если в адаптивной верстке использовать относительные значения в нужных местах, можно значительно сократить CSS код страницы. Ниже представлены примеры.

Относительный margin

Пример верстки вложенных комментариев, где вместо абсолютных значений используются относительные.

Absolute px margin-left



Comment

Cras ultricies cursus nisi, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing uma. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque.



Comment

Cras ultricies cursus nisi, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing uma. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque.



Comm

Cras ultricies cursus nisi, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing uma. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque

Relative % margin-left



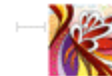
Comment

Cras ultricies cursus nisi, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing uma. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque.



Comment

Cras ultricies cursus nisi, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing uma. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque.



Comment

Cras ultricies cursus nisi, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing uma. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque.

Относительный размер шрифта

При использовании относительных значений (em или %) шрифта наследуются также относительные значения межстрочного пространства и отступов:

`font-size: 100%` →

Heading (normal)

Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras

`font-size: 80%` →

Heading (smaller)

Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras id nibh neque

`font-size: 120%` →

Heading (bigger)

Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras

Относительный padding

Absolute px padding

Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing urna. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque. Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci.

Relative % padding

Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci. Aenean at adipiscing urna. Suspendisse potenti. Suspendisse vitae tellus a neque commodo rutrum. Praesent ac neque vitae massa pharetra pellentesque. Cras ultricies cursus nisl, eget congue tellus consequat nec. Cras id nibh neque, eu dignissim orci.

Перенос слов

При помощи CSS можно переносить непереносимые текстовые конструкции:

```
.break-word { word-wrap: break-word; }
```

Without break-word

`http://www.webdesignerwall.com_title_with_a_long_url_continue_here`

With break-word

`http://www.webdesignerwall.c
om_title_with_a_long_url_con
tinue_here`



Сухой корм Пан Пес Чемпион для собак всех пород со вкусом курицы 10 кг (4820111140275)

344 грн 23 отзыва



Сухой корм для собак Hubertus Gold Adult

1 365 грн 9 отзывов



ТОП ПРОДАЖ



Сухой корм для взрослых собак больших пород Purina Dog Chow Adult Large Breed со вкусом индейки 14 кг (7613034487926)

859 грн 4 отзыва



Сухой корм для щенков и юниоров Josera Kids

1 740 грн

1 566 грн спеццена по промокоду
 20 отзывов
другие предложения (1)



АКЦИЯ



Сухой корм для собак Bosch HPC Mini Adult Ягненок и рис

1 514 грн 6 отзывов



Сухой корм для собак всех пород ACANA Grasslands Dog

3 314 грн
2 816 грн спеццена по промокоду
 3 отзыва



Сухой корм для собак Baskerville Sensitive

1 430 грн 4 отзыва



АКЦИЯ



Сухой корм Nutra Mix Dog Professional для взрослых собак с повышенной физической нагрузкой

783 грн Оставить отзыв



Сухой корм для взрослых собак мелких пород ACANA Adult Small Breed

712 грн
605 грн спеццена по промокоду
 6 отзывов



НОВИНКА



Сухой корм для взрослых активных собак Клуб 4 лапы Премиум Актив 14 кг (4820083909559)

858 грн Оставить отзыв





Wild Horses, The Silk Road Mountain...
PEDaled | 19.3K plays



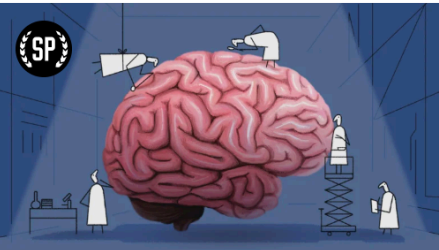
Lea Porcelain - The Love (Official Mu...
Jakob Schmidt | 10.8K plays



Fitting
Emily Avila | 12.7K plays



Accident, MD
Dan Rybicky | 34.1K plays



Deepness Of The Fry
The Animation Workshop | 13.6K plays



Giangrande - Free to Roam [Official V...
gianluca maruotti | 34K plays



ROOM 140
Priscilla González Sainz | 17.3K plays



ANNA OF THE NORTH - LEANING O...
Noah Lee | 16K plays

Mobile first

Одни из самых важных требований в Mobile First разработке это:

- Показать самое важное содержание в первую очередь
- Вебсайт должен быть легковесным и оптимизированным, т.к. скорость подключения мобильной сети может быть слабой в зависимости от местонахождения пользователя
- Вебсайт не должен загружать больше ресурсов, чем требуется пользователю для получения нужной информации, т.к. мобильный Интернет все еще остается дорогим. Дополнительная информация должна грузиться только по требованию пользователя

Плюсы Mobile First подхода

- Один вебсайт для всех устройств
- Пользователи получают важное содержание страницы в первую очередь
- Быстрая загрузка страниц при низкой скорости подключения
- Удобный интерфейс для навигации в мобильном экране
- Минимальное количество веб ресурсов требуемое для отображения основного содержимого – экономия мобильного Интернет трафика

Важная часть содержимого

Обычно на странице много данных. И необходимо решить, что из этого важно в первую очередь для пользователя.

Подробный список данных на странице:

- Главная картинка
 - Основной текст (заголовок, описание)
 - Цена (если имеется)
 - Список характеристик
 - Социальные сервисы Facebook, Twitter, g+
 - Список похожих товаров/услуг
-
- **Иногда требуется скрыть на мобильном устройстве информацию, которая отображается на ПК и планшетах**

Порядок отображения

<https://codepen.io/luiscarvalho/pen/zBazkZ>

Позиционирование

Свойство `position` позволяет точно задать новое местоположение блока относительно того места, где он находился бы в нормальном потоке документа.

По умолчанию все элементы располагаются последовательно один за другим в том порядке, в котором они определены в структуре HTML-документа. Свойство не наследуется (наследуется только с `inherit`).

Static Значение по умолчанию, означает отсутствие позиционирования. Элементы отображаются последовательно один за другим в том порядке, в котором они определены в HTML-документе. Используется для очистки любого другого значения позиционирования.

Relative Относительно позиционированный элемент сдвигается со своего обычного места в разных направлениях относительно границ родительского контейнера, а пространство, которое он занимал, не исчезает.

При этом такой элемент может перекрывать другое содержимое на странице.

Если для относительно позиционированного элемента одновременно задать свойства `top` и `bottom` или `left` и `right`, то в первом случае сработает только `top`, во втором — `left`.

Относительное позиционирование позволяет задавать `z-index` для элемента, а также абсолютно позиционировать дочерние элементы внутри блока.

Absolute Абсолютно позиционированный элемент полностью удаляется из потока документа и позиционируется относительно границ его блока-контейнера (другого элемента или окна браузера). Блок-контейнер для абсолютно позиционированного элемента — ближайший элемент-предок, значение свойства `position` которого не равно `static`.

Местоположение краёв элемента определяется с помощью свойств смещения. Пространство, которое занимал такой элемент, схлопывается, как будто элемента не существовало на странице. Абсолютно позиционированный элемент может перекрывать другие элементы или быть перекрытым ими (за счёт свойства `z-index`).

Fixed Фиксирует элемент в указанном месте страницы. Блок-контейнером фиксированного элемента является окно просмотра, то есть элемент всегда фиксируется относительно окна браузера и не меняет своего положения во время прокрутки страницы. Сам элемент при этом полностью удаляется из основного потока документа и создаётся в новом потоке документа.

Initial Устанавливает значение свойства в значение по умолчанию.

Inherit Наследует значение свойства от родительского элемента.

position: sticky

- <https://codepen.io/enatario/pen/aaXpJd>
- <https://codepen.io/rachelandrew/pen/NbPamG>
- <https://caniuse.com/#search=position%3Asticky>

Элемент с `position: sticky`; фиксируется в рамках ближайшего родителя, когда расстояние до границы ближайшего прокручиваемого родителя достигает указанного в свойствах `top`, `right`, `bottom`, `left` значения.

На то, в каком месте элемент прикрепится и где отцепиться, также влияют свойства `padding`, `margin` и `transform`. Размещение элемента над другими элементами правится с помощью свойства `z-index`.

<http://shpargalkablog.ru/2017/04/position-sticky.html>

Свойства смещения

Задаются для элементов, для которых значение свойства position не равно static.

Могут принимать как положительные, так и отрицательные значения.

`top, right, bottom, left`

`auto` Значение по умолчанию. Вычисляемое значение свойства равно нулю.

`% (px, ...)` Процентные значения вычисляются относительно высоты блока-контейнера для `top` и `bottom` и ширины блока-контейнера для `right` и `left`.

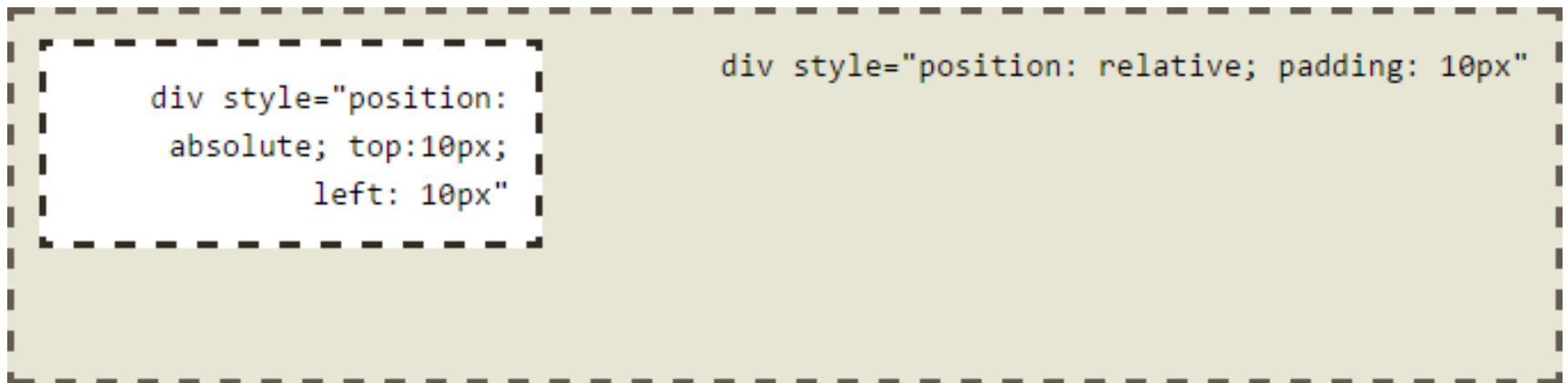
`Initial` Устанавливает значение свойства в значение по умолчанию.

`Inherit` Наследует значение свойства от родительского элемента.

Позиционирование внутри элемента

Для блока-контейнера абсолютно позиционированного элемента задаётся свойство `position: relative` без смещений.

Это позволяет позиционировать элементы внутри элемента-контейнера.



Свойство z-index

Свойство z-index управляет **порядком наложения** позиционированных элементов в случае, когда они накладываются друг на друга.

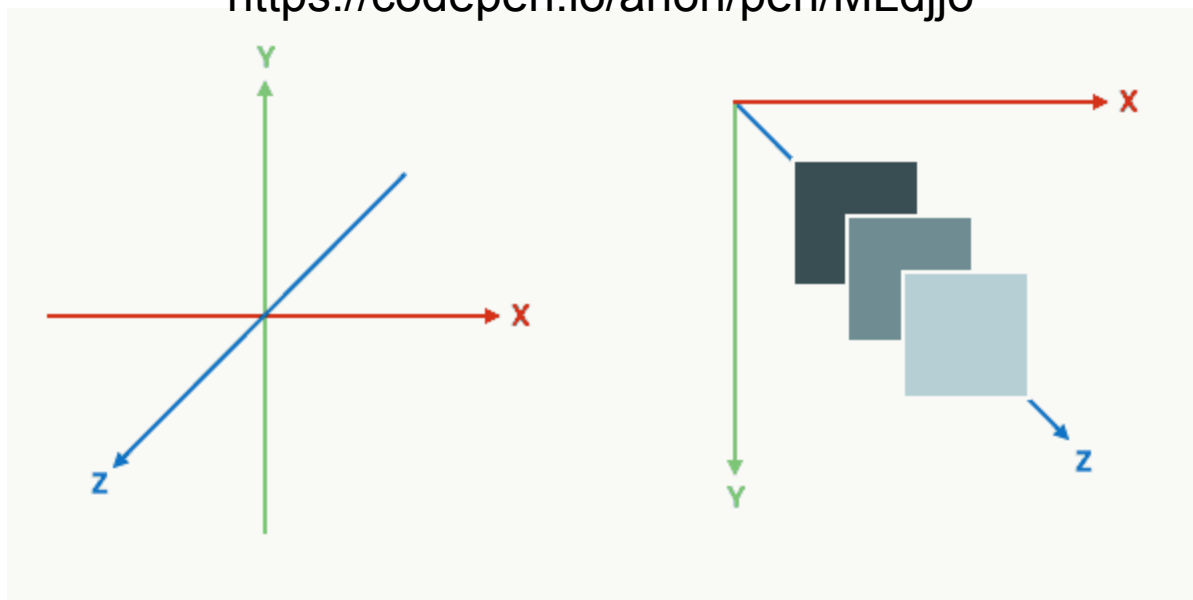
Свойство принимает целые положительные и отрицательные значения.

В нормальном потоке position: static элементы располагаются последовательно один за другим в том порядке, в котором они определены в html-документе.

По умолчанию установлен z-index: auto;.

z-index: число | auto | inherit

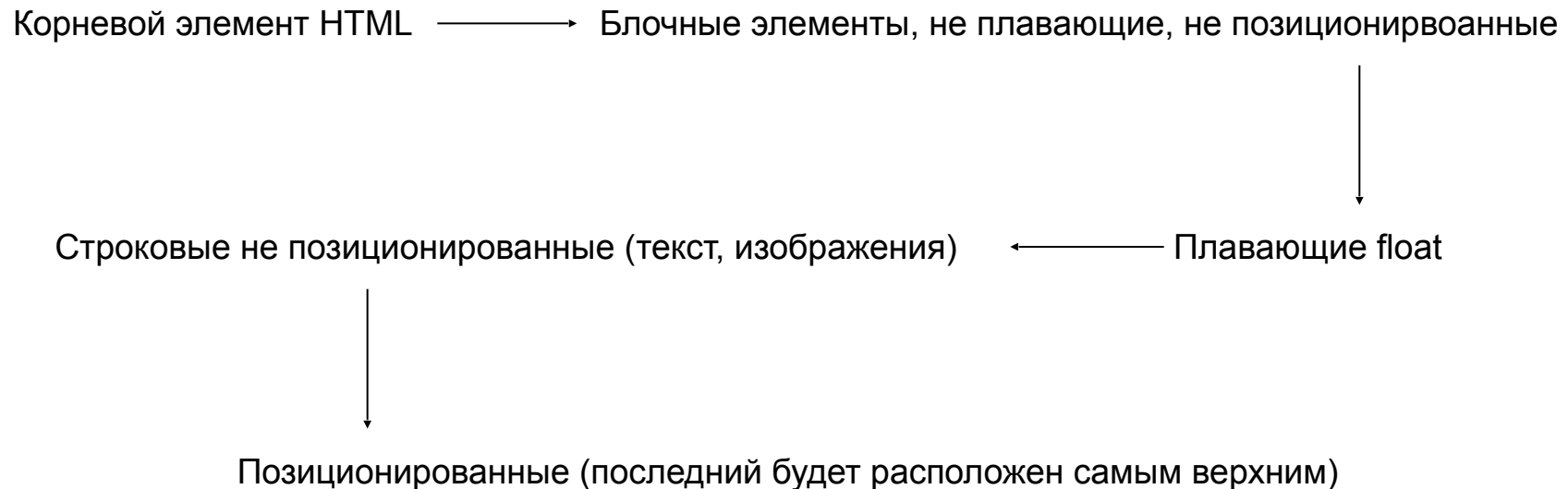
<https://codepen.io/anon/pen/MLdjjo>



Контекст наложения

Если для элементов свойства `z-index` и `position` не заданы явно, контекст наложения равен порядку их расположения в исходном коде.

Если для элементов не задан `z-index`, браузер отображает элементы на странице в следующем порядке:



Кроссбраузерность

Под кроссбраузерностью понимается одинаковое отображение и работа сайта в разных браузерах.

Есть два основных способа добиться одинакового отображения сайта в браузерах:

Разделение стилей с помощью условных комментариев (только для IE)

Разделение стилей для разных браузеров – один из самых распространенных способов кроссбраузерной верстки. Суть метода заключается в идентификации вида браузера пользователя и применение соответствующего стиля совместно с основным.

<!--[if условие]> невидимый **HTML**-код <![endif]-->
<![if условие]> видимый **HTML**-код <![endif]>

Значение	Браузер
IE	Любая версия IE
IE 6	Internet Explorer 6.0
IE 7	Internet Explorer 7.0
IE 8	Internet Explorer 8.0
IE 9	Internet Explorer 9.0

Оператор	Описание	Пример	Комментарий
()	Группирование дополнительных условий. Используется для создания сложных запросов с логическими операторами.	[if !(IE 6) & (lt IE 9)]	Все версии, кроме IE6 и IE9.
!	Логическое НЕ. Условие истинно, если условие следующее за ! не выполняется.	[if !(IE 7)]	Все версии, кроме IE7.
&	Логическое И. Предназначено для объединения нескольких условий. Возвращает true, если все условия выполняются.	[if (gte IE 6) & (lt IE 8)]	IE6, IE7.
	Логическое ИЛИ. Возвращает true, если хотя бы одно из условий выполняется.	[if (IE 6) (IE 7)]	IE6 или IE7.
lt	Оператор «Меньше». Условие истинно, если версия младше указанной.	[if lt IE 9]	Все версии младше IE9.
lte	Оператор «Меньше или равно». Условие истинно, если версия младше указанной или совпадает с ней.	[if lte IE 8]	Все версии младше IE9.
gt	«Больше». Условие истинно, если версия старше указанной.	[if gt IE 7]	Все версии старше IE7.
gte	«Больше или равно». Условие истинно, если версия старше указанной или совпадает с ней.	[if gte IE 7]	IE7, IE8, IE9

<https://habr.com/ru/post/125396/>

Вендорные префиксы

Это приставки (префиксы), используемые производителями (вендорами) браузеров для экспериментальных, еще не принятых в стандарт, CSS-свойств.

Вендорные префиксы представляют собой надпись, которая начитается с «-» или с «_» и для каждого браузера имеет смысл специального маркера, написанного перед CSS свойством.

<http://autoprefixer.github.io/ru/>

Каждый движок, на котором написан браузер, имеет свой вендорный префикс.

Браузеры, которые написанны на движке WebKit, а именно Safari выше третьей версии и GoogleChrome, считывают префикс **-webkit-**,

Safari до третьей версии **-khtml-**, так как имеет в своей основе движок KHTML.

Для Opera можно использовать следующие префиксы: **-o-**, **-op-**, **-xv-**.

Firefox воспримет свойства имеющие приставку **-moz-**, а браузеры корпорации Microsoft те, перед которыми написано **-ms-**.

<https://github.com/postcss/autoprefixer>

<http://autoprefixer.github.io/ru/>

Большинство производителей называют несколько причин, когда нужно использовать вендорные префиксы. Основные, из которых:

- Свойство, которое было написано только для конкретного браузера и не содержится в стандартном списке CSS.
- Свойство ещё разрабатывается или по каким-то причинам не имеет рекомендаций к использованию
- CSS задаёт только часть возможностей стиля.

CSS-хаки

для IE6:

```
.elementstyle {  
  background: #123;  
}
```

или:

```
.elementstyle {  
  -background: #123;  
}
```

или валидно:

```
* html .elementstyle {  
  background: #123;  
}
```

для IE7:

```
*+html .style {  
  background: #123;  
}
```

или валидно:

```
*:first-child+html .elementstyle {  
  background: #123;  
}
```

для IE8:

```
.elementstyle {  
  background: #F12\3/;  
}
```

для Firefox:

```
@-moz-document url-prefix() {  
  .elementstyle {  
    background: #123;  
  }  
}
```

для Safari и Chrome(один движок, помните?):

```
@media screen and (-webkit-min-  
device-pixel-ratio:0) {  
  .elementstyle {  
    background: #123;  
  }  
}
```

или

```
body:last-  
child:not(:root:root) .elementstyle {  
  background: #123;  
}
```