**Основы CSS**

**CSS (Cascading Style Sheets)** — язык таблиц стилей, который позволяет прикреплять стиль (например, шрифты и цвет) к структурированным документам (например, документам HTML и приложениям XML).

Обычно CSS-стили используются для создания и изменения стиля элементов веб-страниц и пользовательских интерфейсов, написанных на языках HTML и XHTML, но также могут быть применены к любому виду XML-документа, в том числе XML, SVG и XUL.

Отделяя стиль представления документов от содержимого документов, CSS упрощает создание веб-страниц и обслуживание сайтов.

CSS поддерживает таблицы стилей для конкретных носителей, поэтому авторы могут адаптировать представление своих документов к визуальным браузерам, слуховым устройствам, принтерам, брайлевским устройствам, карманным устройствам и т.д.

Каскадные таблицы стилей описывают правила форматирования элементов с помощью свойств и допустимых значений этих свойств. Для каждого элемента можно использовать ограниченный набор свойств, остальные свойства не будут оказывать на него никакого влияния.

Объявление стиля состоит из двух частей: **селектора** и **объявления**. В HTML имена элементов нечувствительны к регистру, поэтому «h1» работает так же, как и «H1». Объявление состоит из двух частей: имя свойства (например, color) и значение свойства (grey). Селектор сообщает браузеру, какой именно элемент форматировать, а в блоке объявления (код в фигурных скобках) перечисляются форматирующие команды — свойства и их значения.

РИС. 1. СТРУКТУРА ОБЪЯВЛЕНИЯ

Хотя приведенный пример пытается влиять только на пару свойств, необходимых для рендеринга HTML-документа, он сам по себе квалифицируется как таблица стилей. В сочетании с другими таблицами стилей (одна фундаментальная особенность CSS заключается в том, что таблицы стилей объединяются), правило будет определять окончательное представление документа.

**Виды каскадных таблиц стилей и их специфика**

* **СОДЕРЖАНИЕ:**
* [1. Виды таблиц стилей](https://html5book.ru/osnovy-css/#part1)
* [2. Виды селекторов](https://html5book.ru/osnovy-css/#part2)
* [3. Комбинация селекторов](https://html5book.ru/osnovy-css/#part3)
* [4. Группировка селекторов](https://html5book.ru/osnovy-css/#part4)
* [5. Наследование и каскад](https://html5book.ru/osnovy-css/#part5)

**1. Виды таблиц стилей**

**1.1. Внешняя таблица стилей**

**Внешняя таблица стилей** представляет собой текстовый файл с расширением .css, в котором находится набор CSS-стилей элементов. Файл создаётся в редакторе кода, так же как и HTML-страница. Внутри файла могут содержатся только стили, без HTML-разметки. Внешняя таблица стилей подключается к веб-странице с помощью элемента <link>, расположенного внутри раздела <head></head>. Такие стили работают для всех страниц сайта.

К каждой веб-странице можно присоединить несколько таблиц стилей, добавляя последовательно несколько элементов <link>, указав в атрибуте media назначение данной таблицы стилей. rel="stylesheet" указывает тип ссылки (ссылка на таблицу стилей).

<head>

<link rel="stylesheet" href="css/style.css">

<link rel="stylesheet" href="css/assets.css" media="all">

</head>

HTML

Атрибут type="text/css" не является обязательным по стандарту HTML5, поэтому его можно не указывать. Если атрибут отсутствует, по умолчанию используется значение type="text/css".

**1.2. Внутренние стили**

**Внутренние стили** встраиваются в раздел <head></head> HTML-документа и определяются внутри элемента <style></style>. Внутренние стили имеют приоритет над внешними, но уступают встроенным стилям (заданным через атрибут style).

<head>

<style>

h1,

h2 {

color: red;

font-family: "Times New Roman", Georgia, Serif;

line-height: 1.3em;

}

</style>

</head>

<body>

...

</body>

HTML

**1.3. Встроенные стили**

Когда мы пишем **встроенные стили**, мы пишем CSS-код в HTML-файл, непосредственно внутри элемента с помощью атрибута style:

<p style="font-weight: bold; color: red;">Обратите внимание на этот текст.</p>

HTML

Такие стили действуют только на тот элемент, для которого они заданы.

**1.4. Правило @import**

**Правило @import** позволяет загружать внешние таблицы стилей. Чтобы директива @import работала, она должна располагаться в таблице стилей (внешней или внутренней) перед всеми остальными правилами:

<style>

@import url(mobile.css);

p {

font-size: 0.9em;

color: grey;

}

</style>

HTML

Правило @import также используется для подключения веб-шрифтов:

@import url(https://fonts.googleapis.com/css?family=Open+Sans&subset=latin,cyrillic);

CSS

**2. Виды селекторов**

**Селекторы** представляют структуру веб-страницы. С их помощью создаются правила для форматирования элементов веб-страницы. Селекторами могут быть элементы, их классы и идентификаторы, а также псевдоклассы и псевдоэлементы.

**2.1. Универсальный селектор**

Соответствует любому HTML-элементу. Например, \* {margin: 0;} обнулит внешние отступы для всех элементов сайта. Также селектор может использоваться в комбинации с псевдоклассом или псевдоэлементом: \*:after {CSS-стили}, \*:checked {CSS-стили}.

**2.2. Селектор элемента**

Селекторы элементов позволяют форматировать все элементы данного типа на всех страницах сайта. Например, h1 {font-family: Lobster, cursive;} задаст общий стиль форматирования всех заголовков h1.

**2.3. Селектор класса**

Селекторы класса позволяют задавать стили для одного и более элементов с одинаковым именем класса, размещенных в разных местах страницы или на разных страницах сайта. Например, для создания заголовка с классом headline необходимо добавить атрибут class со значением headline в открывающий тег <h1> и задать стиль для указанного класса. Стили, созданные с помощью класса, можно применять к другим элементам, не обязательно данного типа.

<h1 class="headline">Инструкция пользования персональным компьютером</h1>

HTML

.headline {

text-transform: uppercase;

color: lightblue;

}

CSS

Если элемент имеет несколько атрибутов класса, их значения объединяются с пробелами.

<h1 class="headline post-title">Инструкция пользования персональным компьютером</h1>

HTML

**2.4. Селектор идентификатора**

Селектор идентификатора позволяет форматировать **один** конкретный элемент. Значение id должно быть уникальным, на одной странице может встречаться только один раз и должно содержать хотя бы один символ. Значение не должно содержать пробелов.

Нет никаких других ограничений на то, какую форму может принимать id, в частности, идентификаторы могут состоять только из цифр, начинаться с цифры, начинаться с подчеркивания, состоять только из знаков препинания и т. д.

Уникальный идентификатор элемента может использоваться для различных целей, в частности, как способ ссылки на конкретные части документа с использованием идентификаторов фрагментов, как способ нацеливания на элемент при создании сценариев и как способ стилизации конкретного элемента из CSS.

<div id="sidebar"></div>

HTML

#sidebar {

width: 300px;

float: left;

}

CSS

**2.5. Селектор потомка**

Селекторы потомков применяют стили к элементам, расположенным внутри элемента-контейнера. Например, ul li {text-transform: uppercase;} — выберет все элементы li, являющиеся потомками всех элементов ul.

Если нужно отформатировать потомки определенного элемента, этому элементу нужно задать стилевой класс:

* p.first a {color: green;} — данный стиль применится ко всем ссылкам, потомкам абзаца с классом first;
* p .first a {color: green;} — если добавить пробел, то будут стилизованы ссылки, расположенные внутри любого элемента класса .first, который является потомком элемента <p>;
* .first a {color: green;} — данный стиль применится к любой ссылке, расположенной внутри другого элемента, обозначенного классом .first.

**2.6. Дочерний селектор**

Дочерний элемент является прямым потомком содержащего его элемента. У одного элемента может быть несколько дочерних элементов, а родительский элемент у каждого элемента может быть только один. Дочерний селектор позволяет применить стили только если дочерний элемент идёт сразу за родительским элементом и между ними нет других элементов, то есть дочерний элемент больше ни во что не вложен.

Например, p > strong — выберет все элементы strong, являющиеся дочерними по отношению к элементу p.

**2.7. Сестринский селектор**

Сестринские отношения возникают между элементами, имеющими общего родителя. Селекторы сестринских элементов позволяют выбрать элементы из группы элементов одного уровня:

* h1 + p — выберет все первые абзацы, идущие непосредственно за любым элементом <h1>, не затрагивая остальные абзацы;
* h1 ~ p — выберет все абзацы, являющиеся сестринскими по отношению к любому заголовку h1 и идущие сразу после него.

**2.8. Селектор атрибута**

Селекторы атрибутов выбирают элементы на основе имени атрибута или значения атрибута:

* [атрибут] — все элементы, содержащие указанный атрибут, [alt] — все элементы, для которых задан атрибут alt;
* селектор[атрибут] — элементы данного типа, содержащие указанный атрибут, img[alt] — только картинки, для которых задан атрибут alt;
* селектор[атрибут="значение"] — элементы данного типа, содержащие указанный атрибут с конкретным значением, img[title="flower"] — все картинки, название которых содержит слово flower;
* селектор[атрибут~="значение"] — элементы частично содержащие данное значение, например, если для элемента задано несколько классов через пробел, p[class~="feature"] — абзацы, имя класса которых содержит feature;
* селектор[атрибут|="значение"] — элементы, список значений атрибута которых начинается с указанного слова, p[class|="feature"] — абзацы, имя класса которых feature или начинается на feature;
* селектор[атрибут^="значение"] — элементы, значение атрибута которых начинается с указанного значения, a[href^="http://"] — все ссылки, начинающиеся на http://;
* селектор[атрибут$="значение"] — элементы, значение атрибута которых заканчивается указанным значением, img[src$=".png"] — все картинки в формате png;
* селектор[атрибут\*="значение"] — элементы, значение атрибута которых содержит в любом месте указанное слово, a[href\*="book"] — все ссылки, название которых содержит book.

**2.9. Селектор псевдокласса**

Псевдоклассы — это классы, фактически не прикрепленные к HTML-элементам. Они позволяют применить CSS-правила к элементам при совершении события или подчиняющимся определенному правилу.

Псевдоклассы характеризуют элементы со следующими свойствами:

* :link — не посещенная ссылка;
* :visited — посещенная ссылка;
* :hover — любой элемент, по которому проводят курсором мыши;
* :focus — интерактивный элемент, к которому перешли с помощью клавиатуры или активировали посредством мыши;
* :active — элемент, который был активизирован пользователем;
* :valid — поля формы, содержимое которых прошло проверку в браузере на соответствие указанному типу данных;
* :invalid — поля формы, содержимое которых не соответствует указанному типу данных;
* :enabled — все активные поля форм;
* :disabled — заблокированные поля форм, т.е., находящиеся в неактивном состоянии;
* :in-range — поля формы, значения которых находятся в заданном диапазоне;
* :out-of-range — поля формы, значения которых не входят в установленный диапазон;
* :lang() — элементы с текстом на указанном языке;
* :not(селектор) — элементы, которые не содержат указанный селектор — класс, идентификатор, название или тип поля формы — :not([type="submit"]);
* :target — элемент с символом #, на который ссылаются в документе;
* :checked — выделенные (выбранные пользователем) элементы формы.

**2.10. Селектор структурных псевдоклассов**

Структурные псевдоклассы отбирают дочерние элементы в соответствии с параметром, указанным в круглых скобках:

* :nth-child(odd) — нечётные дочерние элементы;
* :nth-child(even) — чётные дочерние элементы;
* :nth-child(3n) — каждый третий элемент среди дочерних;
* :nth-child(3n+2) — выбирает каждый третий элемент, начиная со второго дочернего элемента (+2);
* :nth-child(n+2) — выбирает все элементы, начиная со второго;
* :nth-child(3) — выбирает третий дочерний элемент;
* :nth-last-child() — в списке дочерних элементов выбирает элемент с указанным местоположением, аналогично с :nth-child(), но начиная с последнего, в обратную сторону;
* :first-child — позволяет оформить только самый первый дочерний элемент;
* :last-child — позволяет форматировать последний дочерний элемент;
* :only-child — выбирает элемент, являющийся единственным дочерним элементом;
* :empty — выбирает элементы, у которых нет дочерних элементов;
* :root — выбирает элемент, являющийся корневым в документе — элемент html.

**2.11. Селектор структурных псевдоклассов типа**

Указывают на конкретный тип дочернего элемента:

* :nth-of-type() — выбирает элементы по аналогии с :nth-child(), при этом берёт во внимание только тип элемента;
* :first-of-type — выбирает первый дочерний элемент данного типа;
* :last-of-type — выбирает последний элемент данного типа;
* :nth-last-of-type() — выбирает элемент заданного типа в списке элементов в соответствии с указанным местоположением, начиная с конца;
* :only-of-type — выбирает единственный элемент указанного типа среди дочерних элементов родительского элемента.

**2.12. Селектор псевдоэлемента**

Псевдоэлементы используются для добавления содержимого, которое генерируется с помощью свойства content:

* :first-letter — выбирает первую букву каждого абзаца, применяется только к блочным элементам;
* :first-line — выбирает первую строку текста элемента, применяется только к блочным элементам;
* :before — вставляет генерируемое содержимое перед элементом;
* :after — добавляет генерируемое содержимое после элемента.

**3. Комбинация селекторов**

Для более точного отбора элементов для форматирования можно использовать комбинации селекторов:

* a[href][title] — выберет все ссылки, для которых заданы атрибуты href и title;
* img[alt\*="css"]:nth-of-type(even) — выберет все четные картинки, альтернативный текст которых содержит слово css.

**4. Группировка селекторов**

Один и тот же стиль можно одновременно применить к нескольким элементам. Для этого необходимо в левой части объявления перечислить через запятую нужные селекторы:

h1,

h2,

p,

span {

color: tomato;

background: white;

}

CSS

**5. Наследование и каскад**

Наследование и каскад — два фундаментальных понятия в CSS, которые тесно связаны между собой.

**Наследование** заключается в том, что элементы наследуют свойства от своего родителя (элемента, их содержащего).

**Каскад** проявляется в том, как разные виды таблиц стилей применяются к документу, и как конфликтующие правила переопределяют друг друга.

**5.1. Наследование**

**Наследование** является механизмом, с помощью которого определенные свойства передаются от предка к его потомкам. Спецификацией CSS предусмотрено наследование свойств, относящихся к текстовому содержимому страницы, таких как color, font, letter-spacing, line-height, list-style, text-align, text-indent, text-transform, visibility, white-space и word-spacing. Во многих случаях это удобно, так как не нужно задавать размер шрифта и семейство шрифтов для каждого элемента веб-страницы.

Свойства, относящиеся к форматированию блоков, не наследуются. Это background, border, display, float и clear, height и width, margin, min-max-height и -width, outline, overflow, padding, position, text-decoration, vertical-align и z-index.

**Принудительное наследование**

С помощью ключевого слова inherit можно принудить элемент наследовать любое значение свойства родительского элемента. Это работает даже для тех свойств, которые не наследуются по умолчанию.

**Как задаются и работают CSS-стили**

Стили могут наследоваться от родительского элемента (наследуемые свойства или с помощью значения inherit).

Стили, расположенные в таблице стилей ниже, отменяют стили, расположенные в таблице выше.

К одному элементу могут применяться стили из разных источников. Проверить, какие стили применяются, можно в режиме разработчика браузера. Для этого над элементом нужно щёлкнуть правой кнопкой мыши и выбрать пункт «Посмотреть код» (или что-то аналогичное). В правом столбце будут перечислены все свойства, которые заданы для этого элемента или наследуются от родительского элемента, а также файлы стилей, в которых они указаны, и порядковый номер строки кода.

РИС. 2. РЕЖИМ РАЗРАБОТЧИКА В БРАУЗЕРЕ GOOGLE CHROME

При определении стиля можно использовать любую комбинацию селекторов — селектор элемента, псевдокласса элемента, класса или идентификатора элемента.

<div id="wrap" class="box clear"></div>

HTML

div {border: 1px solid #eee;}

#wrap {width: 500px;}

.box {float: left;}

.clear {clear: both;}

CSS

**5.2. Каскад**

**Каскадирование** — это механизм, который управляет конечным результатом в ситуации, когда к одному элементу применяются разные CSS-правила. Существует три критерия, которые определяют порядок применения свойств — правило !important, специфичность и порядок, в котором подключены таблицы стилей.

**Правило !important**

Вес правила можно задать с помощью ключевого слова !important, которое добавляется сразу после значения свойства, например, span {font-weight: bold!important;}. Правило необходимо размещать в конец объявления перед закрывающей скобкой, без пробела. Такое объявление будет иметь приоритет над всеми остальными правилами. Это правило позволяет отменить значение свойства и установить новое для элемента из группы элементов в случае, когда нет прямого доступа к файлу со стилями.

**Специфичность**

Для каждого правила браузер вычисляет **специфичность селектора**, и если у элемента имеются конфликтующие объявления свойств, во внимание принимается правило, имеющее наибольшую специфичность. Значение специфичности состоит из четырех частей: 0, 0, 0, 0. Специфичность селектора определяется следующим образом:

* для id добавляется 0, 1, 0, 0;
* для class добавляется 0, 0, 1, 0;
* для каждого элемента и псевдоэлемента добавляется 0, 0, 0, 1;
* для встроенного стиля, добавленного непосредственно к элементу — 1, 0, 0, 0;
* универсальный селектор не имеет специфичности.

h1 {color: lightblue;} /\*специфичность 0, 0, 0, 1\*/

em {color: silver;} /\*специфичность 0, 0, 0, 1\*/

h1 em {color: gold;} /\*специфичность: 0, 0, 0, 1 + 0, 0, 0, 1 = 0, 0, 0, 2\*/

div#main p.about {color: blue;} /\*специфичность: 0, 0, 0, 1 + 0, 1, 0, 0 + 0, 0, 0, 1 + 0, 0, 1, 0 = 0, 1, 1, 2\*/

.sidebar {color: grey;} /\*специфичность 0, 0, 1, 0\*/

#sidebar {color: orange;} /\*специфичность 0, 1, 0, 0\*/

li#sidebar {color: aqua;} /\*специфичность: 0, 0, 0, 1 + 0, 1, 0, 0 = 0, 1, 0, 1\*/

CSS

В результате к элементу применятся те правила, специфичность которых больше. Например, если на элемент действуют две специфичности со значениями 0, 0, 0, 2 и 0, 1, 0, 1, то выиграет второе правило.

**Порядок подключённых таблиц**

Вы можете создать несколько внешних таблиц стилей и подключить их к одной веб-странице. Если в разных таблицах будут встречаться разные значения свойств одного элемента, то в результате к элементу применится правило, находящееся в таблице стилей, идущей в списке ниже.

**CSS-цвета**

Опубликовано: 15 мая 2014Обновлено: 2 января 2020[7 Comments](https://html5book.ru/css-colors/#disqus_thread)

Модуль CSS color подробно описывает значения, которые позволяют авторам определять цвета и непрозрачность html-элементов, а также значения свойства color.

**Свойство color**

* **СОДЕРЖАНИЕ:**
* [1. Приоритетные цвета: свойство color](https://html5book.ru/css-colors/#color)
* [2. Значения цвета](https://html5book.ru/css-colors/#color-values)
* [2.1. Основные ключевые слова](https://html5book.ru/css-colors/#basic-color)
* [2.2. Числовые значения цвета](https://html5book.ru/css-colors/#numerical-values)
* [2.2.1. Цвета модели RGB](https://html5book.ru/css-colors/#rgb)
* [2.2.2. Цвета модели RGBA](https://html5book.ru/css-colors/#rgba)
* [2.2.3. Ключевое слово transparent](https://html5book.ru/css-colors/#transparent)
* [2.2.4. HSL-цвета](https://html5book.ru/css-colors/#hsl)
* [2.2.5. HSLA-значения цвета](https://html5book.ru/css-colors/#hsla)
* [2.3. Расширенные ключевые слова цвета](https://html5book.ru/css-colors/#extended-color)
* [2.4. Ключевое слово currentColor](https://html5book.ru/css-colors/#current-color)

**1. Приоритетные цвета: свойство color**

Свойство задаёт цвет шрифта с помощью различных систем цветопередачи. Свойство описывает цвет текстового содержимого элемента. Кроме того, оно используется для предоставления потенциального косвенного значения (currentColor) для любых других свойств, которые принимают значения цвета.

Свойство наследуется.

|  |  |
| --- | --- |
| **color** | |
| Значения: |  |
| цвет | Задаётся с помощью значений цвета. |
| inherit | Наследует значение свойства от родительского элемента. |

**2. Значения цвета**

**2.1. Основные ключевые слова**

Список основных ключевых слов включает в себя следующие значения:

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **HEX** | **RGB** | **Цвет** |
| black | #000000 | 0,0,0 |  |
| silver | #C0C0C0 | 192,192,192 |  |
| gray | #808080 | 128,128,128 |  |
| white | #FFFFFF | 255,255,255 |  |
| maroon | #800000 | 128,0,0 |  |
| red | #FF0000 | 255,0,0 |  |
| purple | #800080 | 128,0,128 |  |
| fuchsia | #FF00FF | 255,0,255 |  |
| green | #008000 | 0,128,0 |  |
| lime | #00FF00 | 0,255,0 |  |
| olive | #808000 | 128,128,0 |  |
| yellow | #FFFF00 | 255,255,0 |  |
| navy | #000080 | 0,0,128 |  |
| blue | #0000FF | 0,0,255 |  |
| teal | #008080 | 0,128,128 |  |
| aqua | #00FFFF | 0,255,255 |  |

Названия цветов не чувствительны к регистру.

**Синтаксис**

color: teal;

CSS

**2.2. Числовые значения цвета**

**2.2.1. Цвета модели RGB**

Формат значения RGB в шестнадцатеричном формате — это знак #, за которым сразу следуют три или шесть шестнадцатеричных символов. Трехзначная запись RGB #rgb преобразуется в шестизначную форму #rrggbb путем копирования цифр, а не путем добавления нулей. Например, #fb0 расширяется до #ffbb00. Это гарантирует, что белый #ffffff может быть указан в короткой записи #fff, и удаляет любые зависимости от глубины цвета дисплея.

Формат значения RGB в функциональной нотации — rgb(, за которым следует разделенный запятыми список из трех числовых значений (либо трех целочисленных значений, либо трех процентных значений), за которыми следует символ ). Целочисленное значение 255 соответствует 100% и F или FF в шестнадцатеричной записи:

rgb (255,255,255) = rgb (100%, 100%, 100%) = #FFF

Символы пробела допускаются вокруг числовых значений.

Все цвета [RGB](https://html5book.ru/funkciya-rgb/) указываются в цветовом пространстве sRGB. Пользовательские агенты могут различаться в точности, с которой они представляют эти цвета, но использование sRGB дает однозначное и объективно измеримое определение того, каким должен быть цвет.

Значения за пределами диапазона устройства должны быть обрезаны или отображены в известном диапазоне: значения красного, зеленого и синего необходимо изменить, чтобы они попадали в диапазон, поддерживаемый устройством. Некоторые устройства, например принтеры, имеют диапазоны, отличные от sRGB, поэтому некоторые цвета за пределами диапазона 0..255 sRGB будут представимы (внутри диапазона устройства) и будут отображаться.

**Синтаксис**

color: #fb0;

color: #ffbb00;

color: rgb(255,0,0);

color: rgb(100%, 0%, 0%);

CSS

**2.2.2. Цвета модели RGBA**

Цветовая модель RGB расширена в этой спецификации, чтобы включить alpha, которая управляющая непрозрачностью цвета. В отличие от значений RGB, для значения RGBA нет шестнадцатеричной записи.

Формат значения [RGBA](https://html5book.ru/transparency/#rgba) в функциональной нотации — rgba( за которым следует разделенный запятыми список из трех числовых значений (либо трех целочисленных значений, либо трех процентных значений), за которыми следует значение непрозрачности, а затем ). Целочисленное значение 255 соответствует 100%, rgba (255,255,255,0.8) = rgba (100%,100%,100%,0.8). Символы пробела допускаются вокруг числовых значений.

Параметр непрозрачности применяется ко всему объекту. Любые значения за пределами диапазона от 0.0 (полностью прозрачный) до 1.0 (полностью непрозрачный) будут ограничены этим диапазоном.

**Синтаксис**

color: rgba(0,0,255,0.5);

color: rgba(100%, 50%, 0%, 0.1);

CSS

**2.2.3. Ключевое слово transparent**

Это ключевое слово можно считать сокращением для прозрачного черного цвета rgba (0,0,0,0), которое является его вычисленным значением.

**Синтаксис**

color: transparent;

CSS

**2.2.4. HSL-цвета**

Цвета RGB не интуитивно понятны. CSS3 добавляет числовые цвета hue-saturation-lightness (HSL) в дополнение к числовым цветам RGB. [HSL-цвета](https://html5book.ru/funkciya-hsl/) симметричны свету и темноте, и преобразование HSL в RGB максимально просто.

Цвета HSL кодируются как тройка (оттенок, насыщенность, яркость). Оттенок представлен как угол цветного круга (то есть радуга, представленная в круге). Этот угол обычно измеряется в градусах, так что эта единица измерения неявна в CSS; синтаксически дается только число. По определению красный = 0 = 360, а остальные цвета распределены по кругу, поэтому зеленый = 120, синий = 240 и т.д. Насыщенность и яркость представлены в процентах. 100% — это полное насыщение, а 0% — это оттенок серого. Яркость 0% — черная, 100% — белая, а 50% — нормальная.

**Синтаксис**

color: hsl(0, 100%, 50%);

color: hsl(120, 100%, 50%);

CSS

**2.2.5. HSLA-значения цвета**

Так же, как функциональная нотация rgb() имеет альфа-аналог rgba(), функциональная нотация hsl() имеет альфа-аналог hsla().

Формат значения цвета [HSLA](https://html5book.ru/transparency/#hsla) в функциональной нотации — hsla(, за которым следуют оттенок в градусах, насыщенность и яркость в процентах, и значение непрозрачности, после которого следует символ ). Символы пробела допускаются вокруг числовых значений.

**Синтаксис**

color: hsla(240, 100%, 50%, 0.5);

color: hsla(30, 100%, 50%, 0.1);

CSS

**2.3. Расширенные ключевые слова цвета**

В таблице ниже представлен список цветов, поддерживаемых популярными браузерами.

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **HEX** | **RGB** | **Цвет** |
| aliceblue | #F0F8FF | 240,248,255 |  |
| antiquewhite | #FAEBD7 | 250,235,215 |  |
| aqua | #00FFFF | 0,255,255 |  |
| aquamarine | #7FFFD4 | 127,255,212 |  |
| azure | #F0FFFF | 240,255,255 |  |
| beige | #F5F5DC | 245,245,220 |  |
| bisque | #FFE4C4 | 255,228,196 |  |
| black | #000000 | 0,0,0 |  |
| blanche­dalmond | #FFEBCD | 255,235,205 |  |
| blue | #0000FF | 0,0,255 |  |
| blueviolet | #8A2BE2 | 138,43,226 |  |
| brown | #A52A2A | 165,42,42 |  |
| burlywood | #DEB887 | 222,184,135 |  |
| cadetblue | #5F9EA0 | 95,158,160 |  |
| chartreuse | #7FFF00 | 127,255,0 |  |
| chocolate | #D2691E | 210,105,30 |  |
| coral | #FF7F50 | 255,127,80 |  |
| corn­flowerblue | #6495ED | 100,149,237 |  |
| cornsilk | #FFF8DC | 255,248,220 |  |
| crimson | #DC143C | 220,20,60 |  |
| cyan | #00FFFF | 0,255,255 |  |
| darkblue | #00008B | 0,0,139 |  |
| darkcyan | #008B8B | 0,139,139 |  |
| dark­goldenrod | #B8860B | 184,134,11 |  |
| darkgray | #A9A9A9 | 169,169,169 |  |
| darkgreen | #006400 | 0,100,0 |  |
| darkkhaki | #BDB76B | 189,183,107 |  |
| darkmagenta | #8B008B | 139,0,139 |  |
| darkolivegreen | #556B2F | 85,107,47 |  |
| darkorange | #FF8C00 | 255,140,0 |  |
| darkorchid | #9932CC | 153,50,204 |  |
| darkred | #8B0000 | 139,0,0 |  |
| darksalmon | #E9967A | 233,150,122 |  |
| dark­seagreen | #8FBC8F | 143,188,143 |  |
| darkslate­blue | #483D8B | 72,61,139 |  |
| darkslate­gray | #2F4F4F | 47,79,79 |  |
| dark­turquoise | #00CED1 | 0,206,209 |  |
| darkviolet | #9400D3 | 148,0,211 |  |
| deeppink | #FF1493 | 255,20,147 |  |
| deepsky­blue | #00BFFF | 0,191,255 |  |
| dimgray | #696969 | 105,105,105 |  |
| dodgerblue | #1E90FF | 30,144,255 |  |
| firebrick | #B22222 | 178,34,34 |  |
| floral­white | #FFFAF0 | 255,250,240 |  |
| forest­green | #228B22 | 34,139,34 |  |
| fuchsia | #FF00FF | 255,0,255 |  |
| gainsboro | #DCDCDC | 220,220,220 |  |
| ghostwhite | #F8F8FF | 248,248,255 |  |
| gold | #FFD700 | 255,215,0 |  |
| goldenrod | #DAA520 | 218,165,32 |  |
| gray | #808080 | 128,128,128 |  |
| green | #008000 | 0,128,0 |  |
| green­yellow | #ADFF2F | 173,255,47 |  |
| honeydew | #F0FFF0 | 240,255,240 |  |
| hotpink | #FF69B4 | 255,105,180 |  |
| indianred | #CD5C5C | 205,92,92 |  |
| indigo | #4B0082 | 75,0,130 |  |
| ivory | #FFFFF0 | 255,255,240 |  |
| khaki | #F0E68C | 240,230,140 |  |
| lavender | #E6E6FA | 230,230,250 |  |
| lavender­blush | #FFF0F5 | 255,240,245 |  |
| lawngreen | #7CFC00 | 124,252,0 |  |
| lemon­chiffon | #FFFACD | 255,250,205 |  |
| lightblue | #ADD8E6 | 173,216,230 |  |
| lightcoral | #F08080 | 240,128,128 |  |
| lightcyan | #E0FFFF | 224,255,255 |  |
| lightgolden­rodyellow | #FAFAD2 | 250,250,210 |  |
| lightgray | #D3D3D3 | 211,211,211 |  |
| lightgreen | #90EE90 | 144,238,144 |  |
| lightpink | #FFB6C1 | 255,182,193 |  |
| lightsalmon | #FFA07A | 255,160,122 |  |
| light­seagreen | #20B2AA | 32,178,170 |  |
| light­skyblue | #87CEFA | 135,206,250 |  |
| light­slategray | #778899 | 119,136,153 |  |
| lightsteel­blue | #B0C4DE | 176,196,222 |  |
| light­yellow | #FFFFE0 | 255,255,224 |  |
| lime | #00FF00 | 0,255,0 | <="" td="" style="transition: all 0.3s ease-in-out 0s; box-sizing: border-box; margin: 0px; padding: 5px 15px; border: 1px solid rgb(230, 230, 230); font-size: 14.4px; line-height: 1.4; background: lime;"> |
| limegreen | #32CD32 | 50,205,50 |  |
| linen | #FAF0E6 | 250,240,230 |  |
| magenta | #FF00FF | 255,0,255 |  |
| maroon | #800000 | 128,0,0 |  |
| mediumaqua­marine | #66CDAA | 102,205,170 |  |
| mediumblue | #0000CD | 0,0,205 |  |
| medium­orchid | #BA55D3 | 186,85,211 |  |
| medium­purple | #9370DB | 147,112,219 |  |
| medium­seagreen | #3CB371 | 60,179,113 |  |
| medium­slateblue | #7B68EE | 123,104,238 |  |
| medium­springgreen | #00FA9A | 0,250,154 |  |
| medium­turquoise | #48D1CC | 72,209,204 |  |
| medium­violetred | #C71585 | 199,21,133 |  |
| midnightblue | #191970 | 25,25,112 |  |
| mintcream | #F5FFFA | 245,255,250 |  |
| mistyrose | #FFE4E1 | 255,228,225 |  |
| moccasin | #FFE4B5 | 255,228,181 |  |
| navajo­white | #FFDEAD | 255,222,173 |  |
| navy | #000080 | 0,0,128 |  |
| oldlace | #FDF5E6 | 253,245,230 |  |
| olive | #808000 | 128,128,0 |  |
| olivedrab | #6B8E23 | 107,142,35 |  |
| orange | #FFA500 | 255,165,0 |  |
| orangered | #FF4500 | 255,69,0 |  |
| orchid | #DA70D6 | 218,112,214 |  |
| pale­goldenrod | #EEE8AA | 238,232,170 |  |
| palegreen | #98FB98 | 152,251,152 |  |
| pale­turquoise | #AFEEEE | 175,238,238 |  |
| pale­violetred | #DB7093 | 219,112,147 |  |
| papayawhip | #FFEFD5 | 255,239,213 |  |
| peachpuff | #FFDAB9 | 255,218,185 |  |
| peru | #CD853F | 205,133,63 |  |
| pink | #FFC0CB | 255,192,203 |  |
| plum | #DDA0DD | 221,160,221 |  |
| powder­blue | #B0E0E6 | 176,224,230 |  |
| purple | #800080 | 128,0,128 |  |
| red | #FF0000 | 255,0,0 |  |
| rosybrown | #BC8F8F | 188,143,143 |  |
| royalblue | #4169E1 | 65,105,225 |  |
| saddle­brown | #8B4513 | 139,69,19 |  |
| salmon | #FA8072 | 250,128,114 |  |
| sandybrown | #F4A460 | 244,164,96 |  |
| seagreen | #2E8B57 | 46,139,87 |  |
| seashell | #FFF5EE | 255,245,238 |  |
| sienna | #A0522D | 160,82,45 |  |
| silver | #C0C0C0 | 192,192,192 |  |
| skyblue | #87CEEB | 135,206,235 |  |
| slateblue | #6A5ACD | 106,90,205 |  |
| slategray | #708090 | 112,128,144 |  |
| snow | #FFFAFA | 255,250,250 |  |
| spring­green | #00FF7F | 0,255,127 |  |
| steelblue | #4682B4 | 70,130,180 |  |
| tan | #D2B48C | 210,180,140 |  |
| teal | #008080 | 0,128,128 |  |
| thistle | #D8BFD8 | 216,191,216 |  |
| tomato | #FF6347 | 255,99,71 |  |
| turquoise | #40E0D0 | 64,224,208 |  |
| violet | #EE82EE | 238,130,238 |  |
| wheat | #F5DEB3 | 245,222,179 |  |
| white | #FFFFFF | 255,255,255 |  |
| whitesmoke | #F5F5F5 | 245,245,245 |  |
| yellow | #FFFF00 | 255,255,0 |  |
| yellow­green | #9ACD32 | 154,205,50 |  |

**Синтаксис**

color: gold;

CSS

**2.4. Ключевое слово currentColor**

CSS1 и CSS2 определили начальное значение свойства border-color как значение свойства color, но не определили соответствующее ключевое слово. Это упущение было распознано SVG, поэтому SVG 1.0 ввел значение currentColor для свойств fill, stroke, stop-color, flood-color и lighting-color. CSS3 расширяет значение цвета, добавляя ключевое слово currentColor, чтобы разрешить его использование со всеми свойствами, которые принимают значение color. Это упрощает определение этих свойств в CSS3.

Используемое значение ключевого слова currentColor — это вычисленное значение свойства color. Если ключевое слово установлено в самом свойстве color, оно рассматривается как color: inherit.

**Пример**

body {color: grey;}

div {border: 1px solid currentColor;}

CSS

**CSS-текст**

Модуль **CSS-текст** описывает функции CSS, которые управляют переводом исходного текста в форматированный и переносом строк. Различные свойства CSS обеспечивают контроль над преобразованием регистра, обработкой пробелов, правилами переноса и переносом текста и строк, выравниванием, интервалами и отступами.

Основной единицей текста является символ. Тем не менее, поскольку системы письма не всегда так просты, как основной английский алфавит, то, чем на самом деле является символ, зависит от контекста, в котором используется этот термин. Например, в корейской системе письма каждое квадратное представление слога (например, 한 = хань) можно считать символом. Однако квадратный символ действительно состоит из нескольких букв, каждая из которых представляет фонему (например, ㅎ = h, ㅏ = a, ㄴ = n), и каждая из них также может считаться символом.

CSS построен на [Unicode](https://www.unicode.org/).

* **СОДЕРЖАНИЕ:**
* [1. Преобразование текста: свойство text-transform](https://html5book.ru/css-text/#text-transform)
* [2. Обработка пробелов и переносы строк: свойство white-space](https://html5book.ru/css-text/#white-space)
* [3. Настройка табуляции: свойство tab-size](https://html5book.ru/css-text/#tab-size)
* [4. Разрыв строки и границы слов](https://html5book.ru/css-text/#line-breaking)
* [4.1. Правила разрыва для букв: свойство word-break](https://html5book.ru/css-text/#word-break)
* [4.2. Разрыв строки: свойство line-break](https://html5book.ru/css-text/#line-break)
* [4.3. Расстановка переносов: свойство hyphens](https://html5book.ru/css-text/#hyphens)
* [4.4. Переполнение блока-обертки: свойство overflow-wrap/word-wrap](https://html5book.ru/css-text/#overflow-wrap)
* [5. Выравнивание и выключка строк](https://html5book.ru/css-text/#alignment-justification)
* [5.1. Выравнивание текста: свойство text-align](https://html5book.ru/css-text/#text-align)
* [5.2. Выравнивание текста по умолчанию: свойство text-align-all](https://html5book.ru/css-text/#text-align-all)
* [5.3. Выравнивание последней строки: свойство text-align-last](https://html5book.ru/css-text/#text-align-last)
* [6. Промежутки](https://html5book.ru/css-text/#spacing)
* [6.1. Промежутки между словами: свойство word-spacing](https://html5book.ru/css-text/#word-spacing)
* [6.2. Трекинг: свойство letter-spacing](https://html5book.ru/css-text/#letter-spacing)
* [7. Отступ первой строки: свойство text-indent](https://html5book.ru/css-text/#text-indent)

**1. Преобразование текста: свойство text-transform**

Свойство text-transform стилизует текст. Оно не влияет на базовое содержимое и не должно влиять на содержимое операции копирования и вставки простого текста.

Свойство наследуется.

|  |  |
| --- | --- |
| **text-transform** | |
| Значения: |  |
| none | Значение по умолчанию, означает отсутствие эффектов. |
| capitalize | Изменяет написание первой буквы каждого слова в элементе, делая её прописной. |
| uppercase | Выводит все слова в элементе прописными буквами. |
| lowercase | Выводит все слова в элементе строчными буквами. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

text-transform: none;

text-transform: capitalize;

text-transform: uppercase;

text-transform: lowercase;

text-transform: inherit;

text-transform: initial;

CSS

**2. Обработка пробелов и переносы строк: свойство white-space**

Свойство white-space обрабатывает пробелы между словами и переносы строк внутри элемента.

Свойство наследуется.

|  |  |
| --- | --- |
| **white-space** | |
| Значения: |  |
| normal | Значение по умолчанию. Между словами вставляется только по одному пробелу, дополнительные пробелы отбрасываются. Текст переносится только в случае необходимости. |
| nowrap | Запрещает переносы строк, за исключением применения <br>. |
| pre | Пробелы в тексте не игнорируются, браузер отображает дополнительные пробелы и переносы строк. |
| pre-wrap | Сохраняет пробелы в тексте, делая разрывы строк там, где это необходимо. |
| pre-line | Удаляет лишние пробелы, за исключением случаев <br>. |
| break-spaces | Поведение идентично pre-wrap, за исключением того, что: любая последовательность неудаляемых пробелов всегда занимает место, в том числе в конце строки; возможность переноса строки существует после каждого неудаляемого пробела, в том числе между пробелами. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

white-space: normal;

white-space: nowrap;

white-space: pre;

white-space: pre-wrap;

white-space: pre-line;

white-space: break-spaces;

white-space: inherit;

white-space: initial;

CSS

**3. Настройка табуляции: свойство tab-size**

Свойство tab-size используется для изменения величины отступа, получаемого с помощью клавиши ТAB. Значения свойства игнорируются, когда установлено одно из трёх значений pre-line, normal или nowrap свойства white-space.

Работает только для элементов <textarea> и <pre>, для остальных блочных элементов значение всегда будет равно единице. Значения свойства, указанные в единицах длины, поддерживается только в Chrome 42+.

Свойство наследуется.

|  |  |
| --- | --- |
| **tab-size** | |
| Значения: |  |
| целое число | Любое целое положительное число. По умолчанию табуляция делает отступ, равный восьми пробелам. |
| длина | Значение отступа, указываемое в единицах длины, например, рх. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

tab-size: 0;

tab-size: 10px;

tab-size: inherit;

tab-size: initial;

CSS

**4. Разрыв строки и границы слов**

Когда содержимое на строчном уровне разбивается на строки, оно разбивается на линейные блоки. Такое разбиение называется разрыв строки.

Когда строка прерывается из-за явных элементов управления разрывом строки, например, символа новой строки или тега <br>, начала или конца блока — это принудительный разрыв строки.

Если строка обрывается из-за переноса содержимого, когда браузер создает необязательные разрывы строк, чтобы вписать содержимое — это мягкий перенос.

**4.1. Правила разрыва для букв: свойство word-break**

Свойство word-break определяет возможности мягкого переноса между буквами, т.е. когда допустимо разбивать строки текста. В частности, оно контролирует, существует ли возможность мягкого переноса между смежными типографскими буквенными единицами и/или цифрами. Это не влияет на правила, регулирующие возможности мягкого переноса, созданные пробелами.

Свойство наследуется.

|  |  |
| --- | --- |
| **word-break** | |
| Значения: |  |
| normal | Слова разрываются в соответствии с их обычными правилами. Значение по умолчанию. |
| break-all | Разрыв допускается в пределах слов. Перенос слов не применяется. |
| keep-all | Запрещает разрывы между парами символов. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

word-break: normal;

word-break: break-all;

word-break: keep-all;

word-break: inherit;

word-break: initial;

CSS

**4.2. Разрыв строки: line-break**

Свойство line-break определяет правила переноса строк, применяемых внутри элемента, в частности то, как перенос взаимодействует со знаками препинания и символами.

Свойство наследуется.

|  |  |
| --- | --- |
| **line-break** | |
| Значения: |  |
| auto | Браузер определяет набор используемых ограничений на разрыв строки, которые могут варьироваться в зависимости от длины линии, например, использовать менее строгий набор правил разрыва строки для коротких строк. Значение по умолчанию. |
| loose | Разбивает текст, используя наименее ограничивающий набор правил переноса строк. Обычно используется для коротких строк, например, в газетах. |
| normal | Разбивает текст, используя наиболее распространенный набор правил переноса строк. |
| strict | Разбивает текст, используя строгий набор правил переноса строк. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

line-break: auto;

line-break: loose;

line-break: normal;

line-break: strict;

line-break: inherit;

line-break: initial;

CSS

**4.3. Расстановка переносов: свойство hyphens**

Свойство hyphens определяет, разрешено ли использование переносов для создания возможностей мягкого переноса внутри строки текста.

Расстановка переносов — это контролируемое разбиение слов, при котором им обычно не разрешается разрываться, чтобы улучшить расположение абзацев. Как правило, разбиение слов происходит по слоговым или морфемным границам и при визуальном указании на разделение (обычно путем вставки дефиса, ‐). В некоторых случаях переносы могут также изменить написание слова. В любом случае, перенос слов является только эффектом рендеринга: он не должен влиять ни на содержимое документа, ни на выбор текста или поиск.

CSS Text Level 3 не определяет точные правила переноса слов, поэтому рекомендуется выбирать подходящие для языка точки переноса.

Свойство наследуется.

|  |  |
| --- | --- |
| **hyphens** | |
| Значения: |  |
| none | Слова не переносятся, даже если символы внутри слова явно определяют возможности переноса. |
| manual | Слова переносятся только в тех местах, где внутри слова есть символы, которые явно указывают на возможность переноса слов (специальный символ ­). Значение по умолчанию. |
| auto | Слова могут быть разбиты на возможности переноса, определяемые автоматически соответствующим языку ресурсом переноса в дополнение к тем, которые явно указаны условным дефисом. Необходимо задать язык своего контента (например, используя HTML-атрибут lang или заголовок HTTP Content-Language), чтобы получить правильный автоматический перенос слов. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

hyphens: none;

hyphens: manual;

hyphens: auto;

hyphens: inherit;

hyphens: initial;

CSS

**4.4. Переполнение блока-обертки: свойство overflow-wrap/word-wrap**

Свойство overflow-wrap (или его устаревшее имя word-wrap) указывает, может ли неразрывная строка прерваться в неразрешенных точках, чтобы предотвратить переполнение линейного блока. Работает в том случае, когда свойство white-space разрешает перенос.

Свойство наследуется.

|  |  |
| --- | --- |
| **overflow-wrap, word-wrap** | |
| Значения: |  |
| normal | Неразрывные строки могут разрываться только в разрешенных точках разрыва. Значение по умолчанию. |
| break-word |  |
| anywhere | Неразрывная последовательность символов может быть разбита в произвольной точке, если в строке нет других приемлемых точек разрыва. Влияет только на визуальное отображение, не затрагивая исходный текст. В точке разрыва строки символ переноса не добавляется. Возможности мягкого переноса, представленные в любом месте, учитываются при расчете собственных размеров минимального содержимого. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

overflow-wrap: normal;

overflow-wrap: break-word;

overflow-wrap: anywhere;

overflow-wrap: inherit;

overflow-wrap: initial;

CSS

**5. Выравнивание и выключка строк**

Выравнивание и выключка строк контролируют, как встроенный контент распределяется в линейном блоке.

**5.1. Краткая запись для выравнивания текста: свойство text-align**

Блок текста представляет собой набор линейных блоков. Свойство text-align задает свойства text-align-all и text-align-last и описывает, как блоки на уровне строки в каждом линейном блоке выравниваются относительно начальной и конечной сторон линейного блока. Значения, отличные от justify-all или match-parent, присваиваются text-align-all и сбрасываются в text-align-last на auto.

Свойство наследуется.

|  |  |
| --- | --- |
| **text-align** | |
| Значения: |  |
| start | Содержимое на уровне строки выравнивается по начальному краю линейного блока. Значение по умолчанию. |
| end | Содержимое на уровне строки выравнивается по конечному краю линейного блока. |
| left | Содержимое на уровне строки выравнивается по левому краю строки линейного блока. В вертикальных системах письменности это будет физический верх или низ, в зависимости от ориентации текста. |
| right | Содержимое на уровне строки выравнивается по правому краю строки линейного блока. В вертикальных системах письменности это будет физический верх или низ, в зависимости от ориентации текста. |
| center | Содержимое на уровне строки центрируется внутри линейного блока. |
| justify | Текст выравнивается по ширине линейного блока, чтобы точно заполнить поле строки, прижимаюсь к левому и правому краям родительского элемента. Если иное не указано в text-align-last, последняя строка перед принудительным разрывом или конец блока выравнивается по началу. Пробелы между словами и буквами распределяются таким образом, чтобы длина всех строк была равна. Разные браузеры могут увеличить как отступы между словами, так и интервалы между буквами. |
| justify-all | Устанавливает text-align-all и text-align-last в justify, также выравнивая последнюю строку. |
| match-parent | Значение ведет себя так же, как inherit за исключением того, что унаследованное значение start или end интерпретируется относительно значения direction (или исходного содержащего блока, если нет родителя) и приводит к вычисленному значению left или right. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

text-align: start;

text-align: end;

text-align: left;

text-align: right;

text-align: center;

text-align: justify;

text-align: justify-all;

text-align: match-parent;

text-align: inherit;

CSS

**5.2. Выравнивание текста по умолчанию: свойство text-align-all**

Свойство text-align-all — сокращенный вариант свойства text-align определяет выравнивание всех строк содержимого в контейнере блока, за исключением последних строк, переопределенных значением text-align-last. Принимает значения start, end, left, right, center, justify и match-parent.

Свойство наследуется.

**Синтаксис**

text-align-all: start;

text-align-all: end;

text-align-all: left;

text-align-all: right;

text-align-all: center;

text-align-all: justify;

text-align-all: match-parent;

text-align-all: inherit;

CSS

**5.3. Выравнивание последней строки: свойство text-align-last**

Свойство text-align-last описывает, как выравнивается последняя строка блока или строки непосредственно перед принудительным разрывом строки.

Если задано значение auto, содержимое в соответствующей строке выравнивается по text-align-all, если только для text-align-all не настроено justify — в этом случае оно выравнивается по началу блока. Все остальные значения интерпретируются как описано для text-align.

Принимает значения auto, start, end, left, right, center, justify и match-parent.

Свойство наследуется.

**Синтаксис**

text-align-last: auto;

text-align-last: start;

text-align-last: end;

text-align-last: left;

text-align-last: right;

text-align-last: center;

text-align-last: justify;

text-align-last: match-parent;

CSS

**6. Промежутки**

CSS позволяет контролировать промежутки между словами и типографскими символами с помощью свойств word-spacing и letter-spacing.

**6.1. Промежутки между словами: свойство word-spacing**

Свойство word-spacing определяет дополнительный интервал между словами.  
Устанавливает интервалы между словами. Можно использовать положительные и отрицательные значения. При отрицательном значении слова могут накладываться друг на друга.

На значение word-spacing оказывает влияние значение свойства text-align в случае выравнивания текста по ширине.

Свойство наследуется.

|  |  |
| --- | --- |
| **word-spacing** | |
| Значения: |  |
| normal | Дополнительный интервал не применяется. Вычисляет в 0. Значение по умолчанию. |
| длина | Задает дополнительный интервал в дополнение к внутреннему интервалу между словами, определенному шрифтом. Значения могут быть отрицательными, но могут быть ограничения, зависящие от реализации. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

word-spacing: normal;

word-spacing: 1px;

word-spacing: 0.2em;

word-spacing: 1rem;

word-spacing: inherit;

word-spacing: initial;

CSS

**6.2. Трекинг: свойство letter-spacing**

Свойство letter-spacing определяет дополнительный интервал, или трекинг, между смежными типографскими символами. Межбуквенный интервал является дополнением к [кернингу](https://html5book.ru/css3-shrifty/#font-kerning) и word-spacing. В зависимости от действующих правил выравнивания пользовательские агенты могут дополнительно увеличивать или уменьшать расстояние между типографскими символьными единицами для выравнивания текста.

Свойство наследуется.

|  |  |
| --- | --- |
| **letter-spacing** | |
| Значения: |  |
| normal | Дополнительный интервал не применяется. Вычисляет в 0. Значение по умолчанию. |
| длина | Определяет дополнительный интервал между типографскими символами. Значения могут быть отрицательными, но могут быть ограничения, зависящие от реализации. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

letter-spacing: normal;

letter-spacing: 0.1em;

letter-spacing: 2px;

letter-spacing: inherit;

letter-spacing: initial;

CSS

**7. Отступ первой строки: свойство text-indent**

Свойство text-indent задает отступ, применяемый к строкам встроенного содержимого в блоке. Отступ обрабатывается как поле, примененное к начальному краю линейного блока.  
Если в первой строке блочного элемента присутствует изображение, то оно сдвинется вместе с остальным текстом.

Свойство наследуется.

|  |  |
| --- | --- |
| **text-indent** | |
| Значение: |  |
| длина/% | Размер отступа в виде абсолютной длины. Процентное значение вычисляется от собственной логической ширины блока-контейнера. Значение по умолчанию 0. |
| each-line | Отступы затрагивают первую строку каждого блока-контейнера и каждую строку после принудительного разрыва строки (но не строки после с мягким переносом). |
| hanging | Обратное преобразование. Все строки, кроме первой, будут с отступом. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

text-indent: 5mm;

text-indent: 20px;

text-indent: 5%;

text-indent: 2em each-line;

text-indent: 2em hanging;

text-indent: inherit;

text-indent: initial;

CSS

**CSS блочная модель**

Модуль CSS Box Model описывает свойства padding и margin, которые создают поля внутри и отступы снаружи CSS блока. Размеры блока также могут быть увеличены за счет [рамки](https://html5book.ru/css-border/).

Каждый блок имеет прямоугольную **область содержимого** в центре, **поля вокруг** содержимого, **рамку** вокруг полей и **отступ за пределами** рамки. Размеры этих областей определяют свойства padding и его подсвойства — padding-left, padding-topи т.д., border и его подсвойства, margin и его подсвойства.

**CSS блочная модель в деталях**

* **СОДЕРЖАНИЕ:**
* [1. Определение блочной модели](https://html5book.ru/css-blochnaya-model/#box-model)
* [2. Отступы элемента](https://html5book.ru/css-blochnaya-model/#margins)
* [2.1. Схлопывание вертикальных отступов](https://html5book.ru/css-blochnaya-model/#margins-collapse)
* [2.2. Выпадение вертикальных отступов](https://html5book.ru/css-blochnaya-model/#indentation)
* [2.3. Физические свойства отступов: свойства margin-top, margin-right, margin-bottom, margin-left](https://html5book.ru/css-blochnaya-model/#margin-props)
* [2.4. Краткая запись отступов: свойство margin](https://html5book.ru/css-blochnaya-model/#margin)
* [3. Поля элемента](https://html5book.ru/css-blochnaya-model/#paddings)
* [3.1. Физические свойства полей: свойства padding-top, padding-right, padding-bottom, padding-left](https://html5book.ru/css-blochnaya-model/#padding-props)
* [3.2. Краткая запись полей: свойство padding](https://html5book.ru/css-blochnaya-model/#padding)
* [4. Рамки элемента](https://html5book.ru/css-blochnaya-model/#borders)

**1. Определение блочной модели**

Каждый блок имеет **область содержимого**, в которой находится текст, дочерние элементы, изображение и т.п., и необязательные окружающие ее padding, border и margin. Размер каждой области определяется соответствующими свойствами и может быть нулевым, или, в случае margin, отрицательным.

margin  
padding  
borderобласть содержимого

* край содержимого
* край поля
* край рамки
* край отступа

ФИГУРА 1. ОБЛАСТИ И КРАЯ БЛОКА

Поля, рамка и отступы могут быть разбиты на верхний, правый, нижний и левый сегменты, каждый из которых независимо управляется своим соответствующим свойством.

Фон области содержимого, полей и рамки блока определяется свойствами [фона](https://html5book.ru/css-background/). Область рамки может быть дополнительно окрашена с помощью свойства border. Отступы элемента всегда прозрачны, что позволяет показывать фон родительского элемента.

Так как поля и отступы элемента не являются обязательными, по умолчанию их значение равно нулю. Тем не менее, некоторые браузеры добавляют этим свойствам положительные значения по умолчанию на основе своих таблиц стилей. Очистить стили браузеров для всех элементов можно при помощи универсального селектора:

\* {

margin: 0;

padding: 0;

}

CSS

**2. Отступы элемента**

Отступы окружают край рамки элемента, обеспечивая расстояние между соседними блоками. Свойства отступов определяют их толщину. Применяются ко всем элементам, кроме внутренних элементов таблицы. Сокращенное свойство margin задает отступы для всех четырех сторон, а его подсвойства задают отступ только для соответствующей стороны.

Смежные вертикальные отступы элементов в блочной модели схлопываются.

**2.1. Схлопывание вертикальных отступов**

Смежные вертикальные отступы двух или более элементов уровня блока margin объединяются (перекрываются). При этом ширина общего отступа равна ширине большего из исходных. Исключение составляют отступы корневого элемента, которые не схлопываются.

* Область содержимого

margin-bottom: 30px;

* Область содержимого

margin-top: 15px;

* Область содержимого

margin-bottom: 30px; width: 100%; display: inline-block;

* Область содержимого

margin-top: 15px; width: 100%; display: inline-block;

ФИГУРА 2. СХЛОПЫВАНИЕ ОТСТУПОВ И ОДИН ИЗ ВАРИАНТОВ РЕШЕНИЯ

Объединение отступов выполняется только для блочных элементов в нормальном потоке документа. Если среди схлопывающихся отступов есть отрицательные значения, то браузер добавит отрицательное значение к положительному, а полученный результат и будет расстоянием между элементами. Если положительных отступов нет, то максимум абсолютных значений соседних отступов вычитается из нуля.

Отступы не схлопываются:

* Между плавающим блоком и любым другим блоком;
* У плавающих элементов и элементов со значением overflow, отличным от visible, со своими дочерними элементами в потоке;
* У абсолютно позиционированных элементов, даже с их дочерними элементами;
* У строчно-блочных элементов.

Для предотвращения проблемы схлопывания рекомендуется задавать для всех элементов только верхний или нижний margin.

**2.2. Выпадение вертикальных отступов**

Если внутри одного блока расположить другой блок и задать ему margin-top, то внутренний блок прижмется к верхнему краю родительского, а у родительского элемента появится отступ сверху, т.е. внутренний блок «выпадет» из родительского блока. Если у родительского элемента также был задан верхний отступ, то выберется наибольшее из значений.

Чтобы избавиться от эффекта выпадения, можно задать родительскому элементу padding-top или добавить border-top: 1px solid transparent.

**2.3. Физические свойства отступов: свойства margin-top, margin-right, margin-bottom, margin-left**

Свойства устанавливают верхний, правый, нижний и левый отступ блока элемента соответственно. Отрицательные значения допускаются, но могут существовать ограничения для конкретной реализации.

Свойства не наследуются.

|  |  |
| --- | --- |
| **margin-top/margin-right/margin-bottom/margin-left** | |
| Значения: |  |
| длина | Размер отступа задается в единицах длины, например, px, in, em. Значение по умолчанию 0. |
| % | Вычисляется относительно ширины блока контейнера. Изменяются, если изменяется ширина родительского элемента. |
| auto | Для элементов уровня строки, плавающих (float) значения margin-left или margin-right вычисляются в 0. Если для элементов уровня блока задано margin-left: auto или margin-right: auto — соответствующее поле расширяется до края содержащего блока, если оба — их значения становятся равными, что горизонтально центрирует элемент относительно краев содержащего блока. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

margin-top: 20px;

margin-right: 1em;

margin-bottom: 5%;

margin-left: auto;

margin-top: inherit;

margin-right: initial;

CSS

**2.4. Краткая запись отступов: свойство margin**

Свойство margin является сокращенным свойством для установки margin-top, margin-right, margin-bottom и margin-left в одном объявлении.

Если существует только одно значение, оно применяется ко всем сторонам.

Если два — верхний и нижний отступы устанавливаются на первое значение, а правый и левый — устанавливаются на второе.

Если имеется три значения — верхний отступ устанавливается на первое значение, левый и правый — на второе, а нижний — на третье.

Если есть четыре значения — они применяются сверху, справа, снизу и слева соответственно.

**3. Поля элемента**

Область полей представляет собой пространство между краем области содержимого и рамкой элемента. Свойства полей определяют толщину их области. Применяются ко всем элементам, кроме внутренних элементов таблицы (за исключением ячеек таблицы). Сокращенное свойство padding задает поля для всех четырех сторон, а подсвойства устанавливают только их соответствующие стороны.

Фоны элемента по умолчанию закрашивают поля элемента и пространство под его рамкой. Это поведение можно настроить с помощью свойств [background-origin](https://html5book.ru/css-background/#background-origin) и [background-clip](https://html5book.ru/css-background/#background-clip).

**3.1. Физические свойства полей: свойства padding-top, padding-right, padding-bottom, padding-left**

Свойства устанавливают верхнее, правое, нижнее и левое поля соответственно. Отрицательные значения недопустимы.

Свойства не наследуются.

|  |  |
| --- | --- |
| **padding-top/padding-right/padding-bottom/padding-left** | |
| Значения: |  |
| длина | Поля элемента задаются при помощи единиц длины, например, px, pt, cm. Значение по умолчанию 0. |
| % | Вычисляются относительно ширины родительского элемента, могут меняться при изменении ширины элемента. Поля сверху и снизу равны полям слева и справа, т.е. верхние и нижние поля тоже вычисляются относительно ширины элемента. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

padding-top: 0.5em;

padding-right: 0;

padding-bottom: 2cm;

padding-left: 10%;

padding-top: inherit;

padding-bottom: initial;

CSS

**3.2. Краткая запись полей: свойство padding**

Свойство padding является сокращенным свойством для установки padding-top, padding-right, padding-bottom и padding-left в одном объявлении.

Если существует только одно значение, оно применяется ко всем сторонам.

Если есть два значения, верхнее и нижнее поля устанавливаются на первое значение, а правое и левое — на второе.

Если имеется три значения, верхнее поле устанавливается на первое значение, левое и правое — на второе, а нижнее — на третье.

Если есть четыре значения — они применяются сверху, справа, снизу и слева соответственно.

**4. Рамки элемента**

[Рамки элемента](https://html5book.ru/css-border/) заполняют область рамок, визуально очерчивая края блока. Свойства рамок определяют толщину области границы блока, а также ее стиль и цвет.

**CSS-списки**

**CSS-списки** — набор свойств, отвечающих за оформление списков. Использование HTML-списков очень распространено при создании панелей навигации по сайту. Элементы списка представляют набор блочных элементов.

С помощью стандартных CSS-свойств можно **изменить внешний вид маркера списка**, **добавить изображение для маркера**, а также **изменить местоположение маркера**. Высоту блока маркера можно задать свойством line-height.

**Оформление списков с помощью CSS-стилей**

* **СОДЕРЖАНИЕ:**
* [1. Тип маркера списка list-style-type](https://html5book.ru/css-spiski/#list-style-type)
* [2. Изображения для элементов списка list-style-image](https://html5book.ru/css-spiski/#list-style-image)
* [3. Местоположение маркера списка list-style-position](https://html5book.ru/css-spiski/#list-style-position)
* [4. Краткая форма задания стилей списка list-style](https://html5book.ru/css-spiski/#list-style)
* [5. Примеры красивого оформления списков](https://html5book.ru/css-spiski/#part5)

**1. Тип маркера списка list-style-type**

Свойство изменяет типа маркера или **удаляет маркер** для маркированного и нумерованного списков.

Свойство наследуется.

|  |  |
| --- | --- |
| **list-style-type** | |
| Значения: |  |
| disc | Значение по умолчанию. В качестве маркера элементов списка выступает закрашенный кружок. |
| armenian | Традиционная армянская нумерация. |
| circle | В качестве маркера выступает незакрашенный кружок. |
| cjk-ideographic | Идеографическая нумерация. |
| decimal | 1, 2, 3, 4, 5, … |
| decimal-leading-zero | 01, 02, 03, 04, 05, … |
| georgian | Традиционная грузинская нумерация. |
| hebrew | Традиционная еврейская нумерация. |
| hiragana | Японская нумерация: a, i, u, e, o, … |
| hiragana-iroha | Японская нумерация: i, ro, ha, ni, ho, … |
| katakana | Японская нумерация: A, I, U, E, O, … |
| katakana-iroha | Японская нумерация: I, RO, HA, NI, HO, … |
| lower-alpha | a, b, c, d, e, … |
| lower-greek | Строчные символы греческого алфавита. |
| lower-latin | a, b, c, d, e, … |
| lower-roman | i, ii, iii, iv, v, … |
| none | Маркер отсутствует. |
| square | В качестве маркера выступает закрашенный или незакрашенный квадрат. |
| upper-alpha | A, B, C, D, E, … |
| upper-latin | A, B, C, D, E, … |
| upper-roman | I, II, III, IV, V, … |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

list-style-type: disc;

list-style-type: circle;

list-style-type: square;

list-style-type: decimal;

list-style-type: decimal-leading-zero;

list-style-type: lower-roman;

list-style-type: upper-roman;

list-style-type: lower-greek;

list-style-type: lower-latin;

list-style-type: upper-latin;

list-style-type: armenian;

list-style-type: georgian;

list-style-type: lower-alpha;

list-style-type: upper-alpha;

list-style-type: none;

list-style-type: inherit;

list-style-type: initial;

CSS

РИС. 1. ПРИМЕР ОФОРМЛЕНИЯ МАРКИРОВАННОГО И НУМЕРОВАННОГО СПИСКОВ

**2. Изображения для элементов списка list-style-image**

В качестве маркера элементов списка можно использовать изображения и градиентые заливки.

Свойство наследуется.

|  |  |
| --- | --- |
| **list-style-image** | |
| Значения: |  |
| url(url) | Путь к изображению. |
| none | значение по умолчанию, означает отсутствие изображения. Также убирает изображение для элемента из группы элементов с установленным изображением-маркером. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

list-style-image: url("images/romb.png");

list-style-image: linear-gradient(#FF7A2F 0, #FF7A2F 50%, #FFB214 50%);

list-style-image: none;

list-style-image: inherit;

list-style-image: initial;

CSS

РИС. 2. ОФОРМЛЕНИЕ МАРКИРОВАННОГО СПИСКА С ПОМОЩЬЮ ИЗОБРАЖЕНИЯРИС. 3. ОФОРМЛЕНИЕ МАРКИРОВАННОГО СПИСКА С ПОМОЩЬЮ ГРАДИЕНТА

**3. Местоположение маркера списка list-style-position**

Данное свойство предоставляет возможность располагать маркер вне или внутри содержимого элемента списка.

Свойство наследуется.

|  |  |
| --- | --- |
| **list-style-position** | |
| Значения: |  |
| outside | Значение по умолчанию. Маркер располагается вне блока с текстом. |
| inside | Маркер списка изображается в одном блоке с текстом. Последующие строки текста будут располагаться под значком маркера, т.е. маркер будет обтекаться текстом. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

list-style-position: inside;

list-style-position: outside;

list-style-image: inherit;

list-style-image: initial;

CSS

РИС. 4. ПРИМЕР РАСПОЛОЖЕНИЯ МАРКЕРА ВНУТРИ И СНАРУЖИ БЛОКА СПИСКА

**4. Краткая форма задания стилей списка**

Можно объединить все три свойства форматирования списка в одно с помощью list-style. Значения свойств могут быть расположены в произвольном порядке, а часть значений может быть опущена. Если присутствует одно значение, то другие свойства примут значения браузера по умолчанию.

**Синтаксис**

ul {

list-style: url("images/romb.png") inside;

}

CSS

**CSS-Таблицы**

Спецификация CSS даёт неограниченные возможности для оформления таблиц. По умолчанию таблица и ячейки таблицы не имеют видимых границ и фона, при этом ячейки внутри таблицы не прилегают вплотную друг к другу.

Ширина ячеек таблицы определяется шириной их содержимого, поэтому ширина столбцов таблицы может быть разной. Высота всех ячеек ряда одинаковая и определяется высотой самой высокой ячейки.

**Форматирование таблиц**

* **СОДЕРЖАНИЕ:**
* [1. Границы таблицы border](https://html5book.ru/css3-tables/#part1)
* [2. Как задать ширину и высоту таблицы](https://html5book.ru/css3-tables/#part2)
* [3. Как задать фон таблицы](https://html5book.ru/css3-tables/#part3)
* [4. Столбцы таблицы](https://html5book.ru/css3-tables/#part4)
* [5. Как добавить таблице заголовок](https://html5book.ru/css3-tables/#caption-side)
* [6. Как убрать промежуток между рамками ячеек](https://html5book.ru/css3-tables/#border-collapse)
* [7. Как увеличить промежуток между рамками ячеек](https://html5book.ru/css3-tables/#border-spacing)
* [8. Как скрыть пустые ячейки таблицы](https://html5book.ru/css3-tables/#empty-cells)
* [9. Компоновка макета таблицы с помощью свойства table-layout](https://html5book.ru/css3-tables/#table-layout)
* [10. Лучшие макеты таблиц (**Топ-10 таблиц**)](https://html5book.ru/css3-tables/#top_table)

**1. Границы таблицы border**

Таблица и ячейки внутри неё по-умолчанию отображаются в браузере без видимых границ. **Границы таблицы** задаются свойством border:

table {

border-collapse: collapse;

/\*убираем пустые промежутки между ячейками\*/

border: 1px solid grey;

/\*устанавливаем для таблицы внешнюю границу серого цвета толщиной 1px\*/

}

CSS

**Границы ячеек заголовка** каждого столбца задаются для элемента th:

th {

border: 1px solid grey;

}

CSS

**Границы ячеек** тела таблицы задаются для элемента td:

td {

border: 1px solid grey;

}

CSS

Толщина рамок соседних ячеек не удваивается, поэтому задать границы для всей таблицы можно следующим способом:

th, td {

border: 1px solid grey;

}

CSS

Внешнюю границу таблицы можно выделить, задав ей увеличенную ширину:

table {

border: 3px solid grey;

}

CSS

Границы можно задавать частично:

/\* устанавливаем для таблицы внешнюю границу серого цвета толщиной 3px \*/

table {

border-top: 3px solid grey;

}

/\* задаём для ячейки тела таблицы границу серого цвета толщиной 1px \*/

td {

border-bottom: 1px solid grey;

}

CSS

Подробнее о свойстве border вы можете прочитать [здесь](https://html5book.ru/css-border/).

**2. Как задать ширину и высоту таблицы**

По умолчанию **ширина и высота таблицы** определяется содержимым её ячеек. Если ширина не задана, то она будет равна ширине самого широкого ряда (строки).

**Ширина таблицы и столбцов** задаётся с помощью свойства width. Если для таблицы задано table {width: 100%;}, то ширина таблицы будет равна ширине блока-контейнера, в котором она находится.

Ширину таблицы и столбцов обычно задают в px или %, например:

table {

width: 600px;

}

th {

width: 20%;

}

td:first-child {

width: 30%;

}

CSS

**Высота таблицы** не задается. **Высотой рядов** таблицы можно управлять, добавив верхний и нижний padding для элементов <td> и <th>.

Фиксировать высоту с помощью свойства height не рекомендуется.

th, td {

padding: 10px 15px;

}

CSS

**3. Как задать фон таблицы**

По умолчанию **фон таблицы** и ячеек прозрачный. Если страница или блок, содержащие таблицу, имеют фон, то он будет просвечиваться сквозь таблицу. Если фон задан и для таблицы и для ячеек, то в местах наложения фона таблицы и ячеек будет виден фон только ячеек. В качестве фона для таблицы в целом и её ячеек могут выступать:

* заливка [сплошным цветом](https://html5book.ru/css-background/#part1),
* [градиентная заливка](https://html5book.ru/css3-gradient/),
* [фоновое изображение](https://html5book.ru/css-background/#part2).

**4. Столбцы таблицы**

Модель CSS таблиц ориентирована в основном на строки (ряды), формируемые с помощью элемента <tr>. На практике бывают случаи, когда необходимо специальное форматирование столбцов, которое возможно следующими способами:

* с помощью элемента <col> можно задать фон для любого количества столбцов;
* с помощью селектора table td:first-child, table td:last-child можно задать стили для первого или последнего столбца таблицы (за исключением первой ячейки заголовка таблицы);
* с помощью селектора table td:nth-child(правило отбора столбцов) можно задать стили для любых столбцов таблицы.

Подробнее про элемент <col> вы можете прочитать [здесь](https://html5book.ru/html-table/#part5).

Подробнее про CSS-селекторы вы сможете прочитать [здесь](https://html5book.ru/osnovy-css/#part5).

**5. Как добавить таблице заголовок**

Добавить заголовок в таблицу можно с помощью элемента <caption>, а с помощью свойства caption-side его можно поместить перед таблицей или под ней. Для горизонтального выравнивания текста заголовка применяется свойство text-align.

Свойство наследуется.

|  |  |
| --- | --- |
| **caption-side** | |
| Значения: |  |
| top | Заголовок таблицы располагается над таблицей. Значение по умолчанию. |
| bottom | Располагает заголовок под таблицей. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

<table>

<caption>Таблица № 1</caption>

<tr>

<th>Company</th>

<th>Q1</th>

<th>Q2</th>

<th>Q3</th>

<th>Q4</th>

</tr>

...

</table>

HTML

caption {

caption-side: bottom;

text-align: right;

padding: 10px 0;

font-size: 14px;

}

CSS

РИС. 1. ПРИМЕР ОТОБРАЖЕНИЯ ЗАГОЛОВКА ПОД ТАБЛИЦЕЙ

**6. Как убрать промежуток между рамками ячеек**

Рамки ячеек таблицы по умолчанию разделены небольшим промежутком. Если задать для таблицы border-collapse: collapse, то промежуток уберётся.

Свойство наследуется.

|  |  |
| --- | --- |
| **border-collapse** | |
| Значения: |  |
| separate | Рамки ячеек располагаются раздельно. |
| collapse | Рамки ячеек сливаются в одну, а промежутки между рамками убираются. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

table {

border-collapse: collapse;

}

CSS

РИС. 2. ПРИМЕР ТАБЛИЦ СО СЛИВАЮЩИМИСЯ И РАЗДЕЛЬНЫМИ РАМКАМИ ЯЧЕЕК

**7. Как увеличить промежуток между рамками ячеек**

С помощью свойства border-spacing можно менять расстояние между рамками ячеек. Данное свойство применяется к таблице в целом.

Свойство наследуется.

|  |  |
| --- | --- |
| **border-spacing** | |
| Значения: |  |
| <длина> <длина> | Добавляет промежутки между рамками как по вертикали, так и по горизонтали. Если заданы две длины, то первая всегда определяет горизонтальный промежуток, а вторая — вертикальный. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

table {

border-collapse: separate;

border-spacing: 10px 20px;

}

table {

border-collapse: separate;

border-spacing: 10px;

}

CSS

РИС. 3. ПРИМЕР ТАБЛИЦ С УВЕЛИЧЕННЫМИ ПРОМЕЖУТКАМИ МЕЖДУ РАМКАМИ ЯЧЕЕК

**8. Как скрыть пустые ячейки таблицы**

Свойство empty-cells скрывает или показывает пустые ячейки. Действует только на ячейки, которые не содержат какой-либо контент. Если для ячейки задан фон, а для таблицы задано table {border-collapse: collapse;}, то ячейка не будет скрыта.

Свойство наследуется.

|  |  |
| --- | --- |
| **empty-cells** | |
| Значения: |  |
| show | Рамка и фон пустой ячейки будут отрисовываться так же, как для ячейки таблицы, имеющей содержимое. |
| hide | Если все ячейки строки пусты, то вся строка отображается так, если бы было задано значение display: none. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

<table>

<tr>

<th>Company</th>

<th>Q1</th>

<th>Q2</th>

<th>Q3</th>

</tr>

<tr>

<td>Microsoft</td>

<td>20.3</td>

<td>30.5</td>

<td></td>

</tr>

<tr>

<td>Google</td>

<td>50.2</td>

<td>40.63</td>

<td>45.23</td>

</tr>

</table>

HTML

table {

border: 1px solid #69c;

border-collapse: separate;

empty-cells: hide;

}

th, td {

border: 2px solid #69c;

}

CSS

РИС. 4. ПРИМЕР СКРЫТИЯ ПУСТОЙ ЯЧЕЙКИ ТАБЛИЦЫ

**9. Компоновка макета таблицы с помощью свойства table-layout**

Компоновка макета таблицы определяется одним из двух подходов: фиксированный макет или автоматический макет. Под компоновкой в данном случае подразумевается как распределяется ширина таблицы между шириной ячеек.

Свойство не наследуется.

|  |  |
| --- | --- |
| **table-layout** | |
| Значения: |  |
| auto | Значение по умолчанию. Ширина макета таблицы определяется шириной её содержимого с учетом значений свойств padding-left, padding-right, border-left width плюс одна ширина border-right последней ячейки в ряду, или заданной шириной ячеек и толщиной рамки. Если ширина ячеек не задана явно, они могут быть разной ширины. |
| fixed | Свойство сработает только в том случае, если для таблицы задана ширина. Ширина ячеек будет одинаковой, а содержимое ячеек, которое не помещается в ячейку, будет наползать под содержимое соседней ячейки. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| inherit | Наследует значение свойства от родительского элемента. |

**Синтаксис**

table {

table-layout: fixed;

}

CSS

**10. Лучшие макеты таблиц**

*По материалам статьи*[*Top 10 CSS Table Designs*](http://www.smashingmagazine.com/2008/08/13/top-10-css-table-designs/)*из журнала Smashing Magazine*

[Перейти на страницу с примерами](https://html5book.ru/examples/demo-top10-tables.html)

**1. Горизонтальный минимализм**

Горизонтальные таблицы — это таблицы, текст в которых читается по горизонтали. Каждая сущность представляет собой отдельную строку. Вы можете оформить подобные таблицы в минималистском стиле, поместив двухпиксельную границу под заголовком th.

<table>

<tr><th>Employee</th><th>Salary</th><th>Bonus</th><th>Supervisor</th></tr>

<tr><td>Stephen C. Cox</td><td>$300</td><td>$50</td><td>Bob</td></tr>

<tr><td>Josephin Tan</td><td>$150</td><td>-</td><td>Annie</td></tr>

<tr><td>Joyce Ming</td><td>$200</td><td>$35</td><td>Andy</td></tr>

<tr><td>James A. Pentel</td><td>$175</td><td>$25</td><td>Annie</td></tr>

</table>

HTML

table {

font-family: "Lucida Sans Unicode", "Lucida Grande", Sans-Serif;

font-size: 14px;

background: white;

max-width: 70%;

width: 70%;

border-collapse: collapse;

text-align: left;

}

th {

font-weight: normal;

color: #039;

border-bottom: 2px solid #6678b1;

padding: 10px 8px;

}

td {

color: #669;

padding: 9px 8px;

transition: .3s linear;

}

tr:hover td {

color: #6699ff;

}

CSS

При большом количестве строк такой дизайн таблиц затрудняет их чтение. Для решения этой проблемы можно добавить однопиксельную границу под всеми элементами td.

td {

border-bottom: 1px solid #ccc;

color: #669;

padding: 9px 8px;

transition: .3s linear;

}

/\*остальной код - как в примере выше\*/

CSS

Добавление эффекта :hover для элемента tr облегчит чтение таблиц, оформленных в минималистском стиле. При наведении курсора мыши на ячейку, остальные ячейки той же строки выделяются одновременно, что упрощает процесс отслеживания информации, если таблицы имеют несколько столбцов.

th {

font-weight: normal;

color: #039;

padding: 10px 15px;

}

td {

color: #669;

border-top: 1px solid #e8edff;

padding: 10px 15px;

}

tr:hover td {

background: #e8edff;

}

CSS

**2. Вертикальный минимализм**

Несмотря на то, что подобные таблицы используются редко, тем не менее, вертикально ориентированные таблицы полезны для категоризации или сравнения описания объектов, представленных колонкой. Можно оформить их в минималистском стиле, добавив пробел, разделяющий столбцы.

th {

font-weight: normal;

border-bottom: 2px solid #6678b1;

border-right: 30px solid white;

border-left: 30px solid white;

color: #039;

padding: 8px 2px;

}

td {

border-right: 30px solid white;

border-left: 30px solid white;

color: #669;

padding: 12px 2px;

}

CSS

**3. «Коробочный» стиль**

Наиболее надежным стилем для оформления таблиц всех типов, является так называемый «коробочный» стиль. Достаточно подобрать хорошую цветовую гамму, а затем задать цвет фона для всех ячеек. Не забудьте подчеркнуть различие между строками, установив границы в качестве разделителя.

th {

font-size: 13px;

font-weight: normal;

background: #b9c9fe;

border-top: 4px solid #aabcfe;

border-bottom: 1px solid white;

color: #039;

padding: 8px;

}

td {

background: #e8edff;

border-bottom: 1px solid white;

color: #669;

border-top: 1px solid transparent;

padding: 8px;

}

tr:hover td {

background: #ccddff;

}

CSS

Самое сложное — найти ту цветовую гамму, которая будет гармонично сочетаться с вашим сайтом. Если сайт нагружен по графике и дизайну, то вам будет довольно трудно использовать этот стиль.

table {

font-family: "Lucida Sans Unicode", "Lucida Grande", Sans-Serif;

font-size: 14px;

max-width: 70%;

width: 70%;

text-align: center;

border-collapse: collapse;

border-top: 7px solid #9baff1;

border-bottom: 7px solid #9baff1;

}

th {

font-size: 13px;

font-weight: normal;

background: #e8edff;

border-right: 1px solid #9baff1;

border-left: 1px solid #9baff1;

color: #039;

padding: 8px;

}

td {

background: #e8edff;

border-right: 1px solid #aabcfe;

border-left: 1px solid #aabcfe;

color: #669;

padding: 8px;

}

CSS

**4. Горизонтальная зебра**

Зебра-таблица выглядит довольно привлекательной и удобной. Дополнительный цвет фона может служить в качестве визуальной подсказки для людей при чтении таблицы.

th {

font-weight: normal;

color: #039;

padding: 10px 15px;

}

td {

color: #669;

border-top: 1px solid #e8edff;

padding: 10px 15px;

}

tr:nth-child(2n) {

background: #e8edff;

}

CSS

**5. Газетный стиль**

Для достижения так называемого газетного эффекта, можно применить границы для элементов таблицы и поиграть с ячейками внутри. Легкий, минималистичный газетный стиль может выглядеть так: обыграйте цветовую гамму, добавьте границы, отступы, разные фоны, и эффект :hover при наведении на строку.

table {

border: 1px solid #69c;

}

th {

font-weight: normal;

color: #039;

border-bottom: 1px dashed #69c;

padding: 12px 17px;

}

td {

color: #669;

padding: 7px 17px;

}

tr:hover td {

background: #ccddff;

}

CSS

table {

border: 1px solid #69c;

}

th {

font-weight: normal;

color: #039;

padding: 10px;

}

td {

color: #669;

border-top: 1px dashed white;

padding: 10px;

background:#ccddff;

}

tr:hover td {

background: #99bcff;

}

CSS

table {

border: 1px solid #6cf;

}

th {

font-weight: normal;

font-size: 13px;

color: #039;

text-transform: uppercase;

border-right: 1px solid #0865c2;

border-top: 1px solid #0865c2;

border-left: 1px solid #0865c2;

border-bottom: 1px solid white;

padding: 20px;

}

td {

color: #669;

border-right: 1px dashed #6cf;

padding: 10px 20px;

}

CSS

**6. Фон таблицы**

Если вы ищете быстрый и уникальный способ оформления таблицы, выберите привлекательное изображение или фото, относящиеся к теме таблицы и установите ее фоном таблицы.

table {

background: url("https://html5book.ru/wp-content/uploads/2019/04/suggestions.png") 98% 86% no-repeat;

}

th {

font-weight: normal;

font-size: 14px;

color: #339;

padding: 10px 12px;

background: white;

}

td {

color: #669;

border-top: 1px solid white;

padding: 10px 12px;

background: rgba(51, 51, 153, .2);

transition: .3s;

}

tr:hover td {

background: rgba(51, 51, 153, .1);

}

CSS