

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ  
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет инфокоммуникаций

Кафедра инфокоммуникационных технологий

**ВЕБ-ТЕХНОЛОГИИ В ИНФОКОММУНИКАЦИЯХ**  
**лабораторный практикум**



**Минск 2021**

## Содержание

Лабораторная работа №3 Каскадные таблицы стилей CSS .....	4
Виды таблиц стилей.....	4
Виды селекторов .....	5
Наследование и каскад .....	9
CSS блочная модель .....	11
Определение блочной модели.....	11
Отступы элемента.....	12
Поля элемента .....	13
Рамки элемента .....	14
Блочные и строчные элементы.....	14
Модель визуального форматирования .....	15
Блочные элементы и блочные контейнеры.....	15
Строчные элементы и строчные контейнеры .....	16
Строчно-блочные элементы .....	16
Ширина содержимого: свойство width .....	16
Минимальная и максимальная ширина: свойства min-width и max-width.....	16
Высота содержимого: свойство height.....	17
Минимальная и максимальная высота: свойства min-height и max-height .....	17
Расчет высоты строки: свойства line-height и vertical-align .....	18
Изменение блочной модели: свойство box-sizing .....	20
CSS-позиционирование.....	21
Схемы позиционирования .....	21
Содержащий блок.....	21
Выбор схемы позиционирования: свойство position .....	22
Смещение блока: свойства top, right, bottom, left .....	24
Обтекание: свойство float.....	26
Управление потоком рядом с плавающими элементами: свойство clear.....	27
Определение контекста наложения: свойство z-index .....	27
Контекст наложения .....	28
CSS-текст .....	28
Преобразование текста: свойство text-transform .....	29
Обработка пробелов и переносы строк: свойство white-space.....	29
Настройка табуляции: свойство tab-size .....	30
Разрыв строки и границы слов.....	30
Выравнивание и выключка строк .....	32
Промежутки .....	34
Отступ первой строки: свойство text-indent .....	35
CSS-шрифты .....	35
Насыщенность шрифта: свойство font-weight .....	37
Ширина шрифта: свойство font-stretch .....	37
Начертание шрифта: свойство font-style.....	38
Размер шрифта: свойство font-size .....	39
Сокращенная запись свойств шрифта: свойство font.....	40
CSS-ссылки.....	41
Псевдоклассы состояний гипертекстовых ссылок .....	41
Выборка отдельных ссылок .....	41
Подчеркивание ссылок .....	41
Изображения для ссылок .....	42

Использование фонового изображения.....	42
Ссылки-кнопки.....	42
Примеры оформления ссылок .....	43
CSS-таблицы.....	43
Границы таблицы border .....	43
Как задать фон таблицы .....	44
Столбцы таблицы .....	44
Как добавить таблице заголовков.....	44
Как убрать промежуток между рамками ячеек.....	45
Как увеличить промежуток между рамками ячеек.....	45
Как скрыть пустые ячейки таблицы.....	45
Компоновка макета таблицы с помощью свойства table-layout .....	46
Примеры оформления таблиц .....	47
CSS-списки .....	47
Тип маркера списка list-style-type .....	47
Изображения для элементов списка list-style-image.....	48
Местоположение маркера списка list-style-position .....	49
Краткая форма задания стилей списка .....	49
Примеры оформления списков.....	49
CSS-фон .....	50
Базовый цвет: свойство background-color .....	50
Источник изображения: свойство background-image .....	50
Укладка изображений: свойство background-repeat .....	51
Фиксация изображения: свойство background-attachment .....	51
Позиционирование изображений: свойство background-position.....	52
Область рисования: свойство background-clip .....	53
Область расположения фона: свойство background-origin .....	54
Размер изображений: свойство background-size .....	54
Краткая запись свойств фона: свойство background.....	55
Множественные фоны .....	55
CSS-рамка .....	56
Стиль рамки border-style .....	56
Цвет рамки border-color .....	56
Ширина рамки border-width.....	57
Задание рамки одним свойством .....	57
Задание рамки для одной границы элемента.....	58
CSS content.....	58
Свойство content .....	58
Форматирование кавычек quotes.....	61
CSS-цвета.....	61
Приоритетные цвета: свойство color.....	61
Значения цвета .....	61
Консоль разработчика .....	63
Задание к лабораторной работе №3.....	66

## Лабораторная работа №3 Каскадные таблицы стилей 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.

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 таблиц стилей

#### Внешняя таблица стилей

**Внешняя таблица стилей** представляет собой текстовый файл с расширением `.css`, в котором находится набор CSS-стилей элементов. Файл создается в редакторе кода, также как и HTML-страница. Внутри файла могут содержаться только стили, без HTML-разметки. Внешняя таблица стилей подключается к веб-странице с помощью тега `<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 тегов `<link>`, указав в атрибуте тега `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>
```

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"`.

### Внутренние стили

**Внутренние стили** встраиваются в раздел `<head>...</head>` HTML-документа и определяются внутри тега `<style>...</style>`.

```
<head>
<style>
h1,
h2 {
color: red;
font-family: "Times New Roman", Georgia, Serif;
line-height: 1.3em;
}
</style>
</head>
<body>
...
</body>
```

### Встроенные стили

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

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

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

### 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>
```

Правило `@import` также используется для подключения веб-шрифтов:

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

## 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.

### 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}`.

### 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`.

## 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>
```

```
.headline {  
  text-transform: uppercase;  
  color: lightblue;  
}
```

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

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

## 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>
```

```
#sidebar {  
  width: 300px;  
  float: left;  
}
```

## 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 тега класса `.first`, который является потомком элемента `<p>`;
- `.first a {color: green;}` – данный стиль применится к любой ссылке, расположенной внутри другого элемента, обозначенного классом `.first`.

## 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 идет 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`.

### 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 тегом `<h1>`, не затрагивая остальные абзацы;
- `h1 ~ p` – выберет все абзацы, являющиеся сестринскими по отношению к любому заголовку `h1` и идущие сразу после него.

### 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`.

### Селектор псевдокласса

Псевдоклассы – это классы, фактически не прикрепленные к HTML-тегам. 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.

### Селектор структурных псевдоклассов

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

- `:nth-child(odd)` – нечетные дочерние элементы;
- `:nth-child(even)` – четные 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 тега;
- `:last-child` – позволяет форматировать последний дочерний элемент тега;
- `: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`.

### Селектор структурных псевдоклассов типа

Указывает на конкретный тип дочернего тега:

- `:nth-of-type()` – выбирает элементы по аналогии с `:nth-child()`, при этом берет 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.

### 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.

### 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`.

### 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;
}
```

## 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.

### 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.

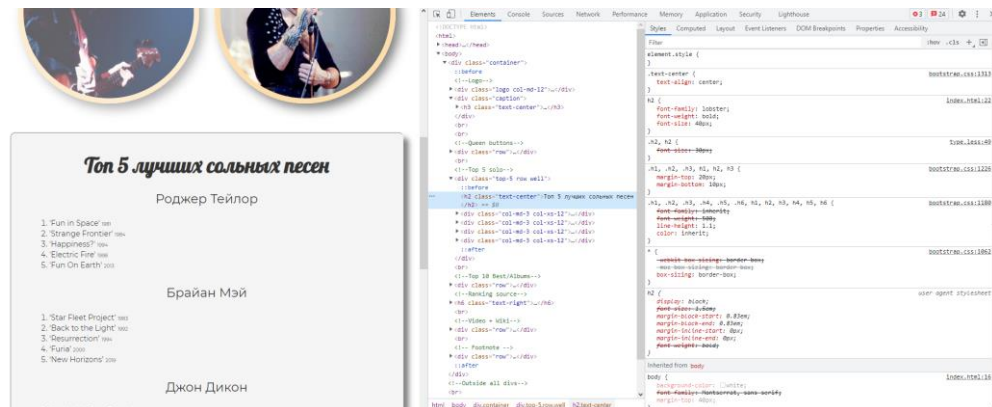
Стили могут наследоваться от родительского элемента (наследуемые свойства или с помощью значения `inherit`).

### Как задаются и работают CSS-стили

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 щелкнуть 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.



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

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

```
div {border: 1px solid #eee;}
#wrap {width: 500px;}
.box {float: left;}
.clear {clear: both;}
```

## 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*/
```

В результате к элементу применяются 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 подключенных 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.

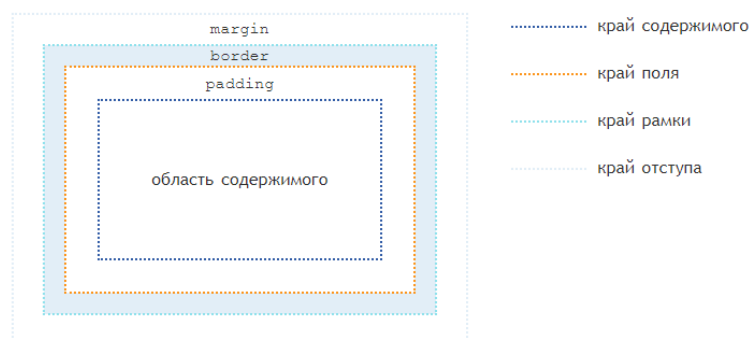
### CSS блочная модель

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.

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-top` и т.д., `border` и его подсвойства, `margin` и его подсвойства.

### 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.



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. 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;  
  box-sizing: border-box;  
}
```

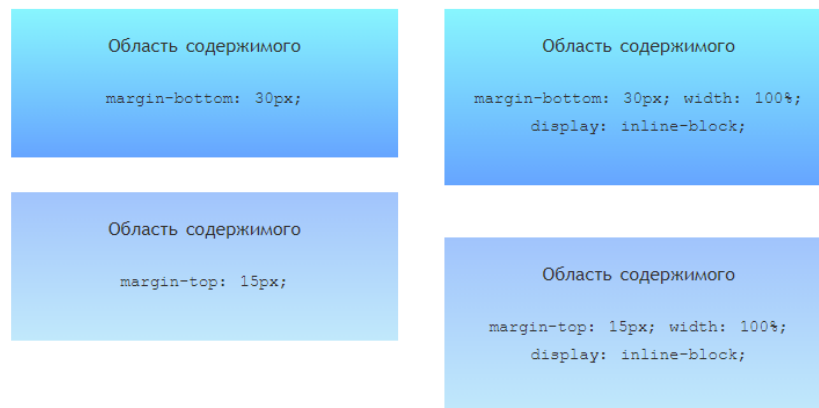
## Отступы элемента

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.

### 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.



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;
- У дочерних элементов `flex`- и `grid`-контейнеров.

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

### 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`.

**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.

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

```
margin-top: 20px;
margin-right: 1em;
margin-bottom: 5%;
margin-left: auto;
margin-top: inherit;
margin-right: initial;
```

**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.

## 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` и `background-clip`.

**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.

Значение	<code>padding-top/padding-right/padding-bottom/padding-left</code>
длина	Размер отступа задается в единицах длины, например, <code>px</code> , <code>pt</code> , <code>cm</code> . Значение по умолчанию <code>0</code> .
%	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.

```
padding-top: 0.5em;  
padding-right: 0;  
padding-bottom: 2cm;  
padding-left: 10%;  
padding-top: inherit;  
padding-bottom: initial;
```

**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.

## Рамки элемента

Рамки элемента заполняют область рамок, визуально очерчивая края блока. Свойства рамок определяют толщину области границы блока, а также ее стиль и цвет.

## Блочные и строчные элементы

Выделяют две основные категории HTML-элементов, которые соответствуют типам их содержимого и поведению в структуре веб-страницы – **блочные** и **строчные** элементы. С помощью блочных элементов можно создавать структуру веб-страницы, строчные элементы используются для форматирования текстовых фрагментов (за исключением элементов `<area>` и `<img>`).

Разделение элементов на блочные и строчные используется в спецификации HTML до версии 4.01. В HTML5 эти понятия заменены более



сложным набором категорий контента, согласно которым каждый HTML-элемент должен следовать правилам, определяющим, какой контент для него допустим.

## Модель визуального форматирования

HTML-документ организован в виде дерева элементов и текстовых узлов. Модель визуального форматирования CSS представляет собой алгоритм, который обрабатывает HTML-документ и выводит его на экран устройства.

Каждый блок в дереве представляет соответствующий элемент или псевдоэлемент, а текст (буквы, цифры, пробелы), находящийся между открывающим и закрывающим тегами, представляет содержимое текстовых узлов.

Чтобы создать дерево блоков, CSS сначала использует каскадирование и наследование, позволяющие назначить вычисленное значение для каждого CSS-свойства каждому элементу и текстовому узлу в исходном дереве.

Затем для каждого элемента CSS генерирует ноль или более блоков в соответствии со значением свойства `display` этого элемента. Как правило, элемент генерирует один основной блок, который представляет самого себя и содержит свое содержимое. Некоторые значения свойства `display`, например, `display: list-item`, генерируют блок основного блока и блок дочернего маркера. Другие, например, `display: none`, приводят к тому, что элемент и/или его потомки вообще не генерируют блоки.

Положение блоков на странице определяется следующими факторами:

- размером элемента (с учетом того, заданы они явно или нет);
- типом элемента (строчный или блочный);
- схемой позиционирования (нормальный поток, позиционированные или плавающие элементы);
- отношениями между элементами в DOM (родительский – дочерний элемент);
- внутренними размерами содержащихся изображений;
- внешней информацией (например, размеры окна браузера).

## Блочные элементы и блочные контейнеры

**Блочные элементы** – элементы высшего уровня, которые форматируются визуально как блоки, располагаясь на странице в окне браузера вертикально. Значения свойства `display`, такие как `block`, `list-item` и `table` делают элементы блочными. Блочные элементы генерируют основной блок, который содержит только блок элемента. Элементы со значением `display: list-item` генерируют дополнительные блоки для маркеров, которые позиционируются относительно основного блока.

`<address>`, `<article>`, `<aside>`, `<blockquote>`, `<dd>`, `<div>`, `<dl>`, `<dt>`, `<details>`, `<fieldset>`, `<figcaption>`, `<figure>`, `<footer>`, `<form>`, `<h1>`–`<h6>`, `<header>`, `<hr>`, `<li>`, `<legend>`, `<nav>`, `<noscript>`, `<ol>`, `<output>`, `<optgroup>`, `<option>`, `<p>`, `<pre>`, `<section>`, `<summary>`, `<table>`, `<ul>`

Блочные элементы могут размещаться непосредственно внутри элемента `<body>`. Они создают разрыв строки перед элементом и после него, образуя прямоугольную область, по ширине занимающую всю ширину веб-страницы или блока-родителя.

Блочные элементы могут содержать как строчные, так и блочные элементы, но не оба типа элементов сразу. При необходимости, строки текста, принадлежащие блочному контейнеру, могут быть обернуты анонимными контейнерами, которые будут вести себя внутри блока как элементы со значением `display: block`, а строчные элементы обернуты элементом `<p>`. Блочные элементы могут содержаться только в пределах блочных элементов.



Элемент `<p>` относится к блочным элементам, но он не должен содержать внутри себя другой элемент `<p>`, а также любой другой блочный элемент.

## Строчные элементы и строчные контейнеры

**Встроенные (строчные) элементы** генерируют внутристрочные контейнеры. Они не формируют новые блоки контента. Значения свойства `display`, такие как `inline` и `inline-table` делают элементы строчными.

`<a>`, `<area>`, `<b>`, `<bdo>`, `<bdi>`, `<cite>`, `<code>`, `<dfn>`, `<del>`, `<em>`, `<i>`, `<iframe>`, `<img>`, `<ins>`, `<kbd>`, `<label>`, `<map>`, `<mark>`, `<s>`, `<samp>`, `<small>`, `<span>`, `<strong>`, `<sub>`, `<sup>`, `<time>`, `<q>`, `<ruby>`, `<u>`, `<var>`

Строчные элементы могут содержать только данные и другие строчные элементы. Исключение составляет элемент `<a>`, который согласно спецификации HTML5 может оборачивать целые абзацы, списки, таблицы, заголовки и целые разделы при условии, что они не содержат другие интерактивные элементы – другие ссылки и кнопки.

## Строчно-блочные элементы

Существует еще одна группа элементов, которые браузер обрабатывает как строчно-блочные `{display: inline-block;}`. Такие элементы являются встроенным, но для них можно задавать поля, отступы, ширину и высоту.

`<audio>`, `<button>`, `<canvas>`, `<embed>`, `<input>`, `<keygen>`, `<meter>`, `<object>`, `<progress>`, `<select>`, `<textarea>`, `<video>`.

## Ширина содержимого: свойство width

Свойство `width` определяет ширину содержимого блока.

Это свойство не применяется к незамещаемым строчным элементам `display: inline;`. Ширина содержимого встроенных блоков определяется шириной отображаемого содержимого внутри них. Встроенные блоки сливаются в линейные блоки. Ширина линейных блоков определяется шириной содержащего блока, но может быть уменьшена из-за наличия свойства `float`.

Отрицательные значения не допускаются.

Свойство не наследуется.

Значение	width
длина	Ширина элемента задается в единицах длины, например, <code>px</code> , <code>em</code> и т.д.
%	Вычисляется относительно ширины содержащего блока. Для абсолютно позиционированных элементов процент вычисляется с учетом ширины области отступов <code>padding</code> содержащего блока.
auto	Ширина вычисляется в зависимости от значений других свойств. Значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
width: 100px;  
width: 10em;  
width: 50%;  
width: auto;  
width: inherit;
```

## Минимальная и максимальная ширина: свойства min-width и max-width

Свойства `min-width` и `max-width` позволяют ограничивать ширину содержимого до определенного диапазона. Значения не могут быть отрицательными. Для `min-width` значение по умолчанию `0`, для `max-width` – `none`.

Свойства не наследуются.

Значение	min-width/max-width
длина	Задаёт фиксированную минимальную или максимальную используемую ширину.
%	Указывает процент для определения используемого значения. Процент рассчитывается относительно ширины содержащего блока.
none	Означает отсутствие ограничений ширины блока.
inherit	Наследует значение свойства от родительского элемента.

```
min-width: 100px;  
min-width: 10em;  
min-width: 50%;  
min-width: inherit;  
  
max-width: 500px;  
max-width: 20em;  
max-width: 80%;  
max-width: none;  
max-width: inherit;
```

### Высота содержимого: свойство height

Свойство `height` определяет высоту содержимого блока. Это свойство не применяется к незамещаемым строчным элементам. Значения длины не могут быть отрицательными.

Свойство не наследуется.

Значение	height
длина	Высота области содержимого задается в единицах длины.
%	Задаёт высоту в процентах. Процент рассчитывается относительно высоты содержащего блока. Если высота содержащего блока не указана явно (то есть зависит от высоты содержимого) и этот элемент не является абсолютно позиционированным, значение вычисляется как <code>auto</code> . Для абсолютно позиционированных элементов процент вычисляется с учетом высоты области отступов <code>padding</code> содержащего блока.
auto	Высота зависит от значений других свойств. Значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
height: 100px;  
height: 10em;  
height: 50%;  
height: auto;  
width: inherit;
```

### Минимальная и максимальная высота: свойства min-height и max-height

Иногда полезно ограничить высоту элементов определенным диапазоном. Свойства `min-height` и `max-height` предлагают эту функциональность.

Свойства не наследуются.

Значение	min-height/max-height
длина	Задаёт фиксированную минимальную или максимальную вычисленную высоту в единицах длины. Значения не могут быть отрицательными.

%	Указывает процент для определения используемого значения. Процент рассчитывается относительно высоты содержащего блока. Если высота содержащего блока не указана явно (т.е. зависит от высоты содержимого) и этот элемент не является абсолютно позиционированным, процентное значение обрабатывается как 0 для <code>min-height</code> или <code>none</code> для <code>max-height</code> .
<code>none</code>	Отсутствие ограничений высоты блока, только для <code>max-height</code> .
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```

min-height: 100px;
min-height: 2em;
min-height: 50%;
min-height: inherit;

max-height: 500px;
max-height: 20em;
max-height: 80%;
max-height: none;
max-height: inherit;

```

### Расчет высоты строки: свойства `line-height` и `vertical-align`

Как описано выше, пользовательские агенты (браузеры) передают блоки встроенного уровня в вертикальный стек линейных блоков. Высота линейного блока определяется следующим образом:

- Высота каждого встроенного прямоугольника в линейном блоке вычисляется. Для замещаемых, `inline-block` и `inline-table` элементов это высота их области поля (`margin box`).
- Блоки уровня строки выравниваются вертикально в соответствии со значением свойства `vertical-align`. Если они выровнены по верху или по низу, они должны быть выровнены так, чтобы минимизировать высоту линейного блока.

Высота линейного блока – это расстояние между самой верхней и самой нижней частью блока. Пустые встроенные элементы генерируют пустые встроенные блоки, но эти блоки по-прежнему имеют поля, отступы, границы, высоту строки и, таким образом, влияют на эти вычисления также, как и элементы с содержимым.

В элементе уровня блока, содержимое которого состоит из элементов встроенного уровня, свойство `line-height` определяет минимальную высоту линейных блоков внутри элемента. Минимальная высота состоит из минимальной высоты над базовой линией и минимальной глубины под ней.

Для элементов уровня строки свойство `line-height` указывает высоту, которая используется при расчете высоты линейного блока.

Отрицательные значения не допустимы.

Свойство наследуется.

Значение	<code>line-height</code>
<code>normal</code>	Сообщает пользовательским агентам установить «разумное» значение на основе шрифта элемента. Значение по умолчанию. Когда элемент содержит текст, отображаемый более чем одним шрифтом, пользовательские агенты могут определить значение <code>normal</code> в соответствии с наибольшим размером шрифта.
длина	Значение задается в единицах длины, создавая фиксированное значение высоты строки. Если задать значение

	меньше единицы, смежные строки будут находить друг на друга.
число	Используемое значение свойства – это число, умноженное на размер шрифта элемента.
%	Вычисленное значение свойства – это процент, умноженный на вычисленный размер шрифта элемента.
inherit	Наследует значение свойства от родительского элемента.

```
line-height: normal;
line-height: 2em;
line-height: 1.5;
line-height: 50%;
line-height: inherit;
```



Свойство `vertical-align` влияет на вертикальное позиционирование в линейном блоке элементов уровня строки: `display: inline` и `display: table-cell`. Значения этого свойства имеют другие значения в контексте таблиц.


Свойство не наследуется.

Значение	<code>vertical-align</code>
baseline	Выравнивает базовую линию элемента по базовой линии его родителя, совмещая среднюю линию элемента со средней линией родительского элемента.
sub	Делает элемент подстрочным (аналогично с тегом <code>&lt;sub&gt;</code> ). Величина понижения элемента может меняться в зависимости от браузера пользователя.
super	Делает элемент надстрочным (аналогично с тегом <code>&lt;sup&gt;</code> ). При этом значения <code>sup</code> и <code>super</code> не меняют размер шрифта, по умолчанию текст надстрочного и подстрочного элемента имеет такой же размер, как и текст родительского элемента.
top	Верхний край элемента совмещается с верхним краем самого высокого элемента в линии.
text-top	Верхний край элемента совмещается с верхним краем шрифта родительского элемента.
middle	Средняя линия элемента (обычно изображения) совмещается с линией, проходящей через середину родительского элемента.
bottom	Нижний край элемента совмещается с нижним краем самого низкого элемента в линии.
text-bottom	Нижний край элемента совмещается с нижним краем шрифта родительского элемента.
%	Не позволяет устанавливать <code>middle</code> , вычисляется как часть <code>line-height</code> элемента, а не его родителя, т.е. если установить значение <code>vertical-align</code> , равное <code>50%</code> для элемента с <code>line-height</code> равным <code>20px</code> , то базовая линия элемента поднимется на <code>10px</code> .
длина	Устанавливает значение в единицах длины, перемещая элемент на заданное расстояние.


inherit


Наследует значение свойства от родительского элемента.


```
vertical-align: baseline;  
vertical-align: sub;  
vertical-align: super;  
vertical-align: text-top;  
vertical-align: text-bottom;  
vertical-align: middle;  
vertical-align: top;  
vertical-align: bottom;  
vertical-align: 6em;  
vertical-align: 10px;  
vertical-align: 25%;  
vertical-align: inherit;
```


Изображение  {vertical-align: baseline} не имеет базовой линии, поэтому оно выравнивается по базовой линии родителя (в данном случае, текста).


Этот абзац содержит **надстрочный** {vertical-align: super} и **подстрочный** {vertical-align: sub} текст, который имеет такой же размер, как и основной текст.

Изображение  {vertical-align: bottom} выравнивается по низу контейнера строки, смещаясь ниже базовой линии текста.

Изображение  {vertical-align: text-bottom} выравнивается относительно нижней линии текста строки.

Изображение  {vertical-align: top} выравнивается по верху контейнера строки, при этом учитывается размер шрифта родительского элемента.

В данном примере изображение  {vertical-align: text-top} выравнивается относительно верхней линии текста строки.

Значение  {vertical-align: middle} выравнивает середину изображения по точке, которая находится на расстоянии, равном половине размера шрифта родительского элемента, над базовой линией текста.

## Изменение блочной модели: свойство box-sizing

Свойство **box-sizing** переключает блочную модель с фиксированных размеров длины и ширины на **content-box** и **border-box**. Это влияет на интерпретацию всех свойств, определяющих размеры, включая **flex-basis**.

Свойство не наследуется.

Значение	box-sizing
content-box	Это поведение ширины и высоты, как указано в CSS2.1. Заданные ширина и высота (и соответствующие min/max-свойства) применяются к ширине и высоте области содержимого элемента. Поля padding и рамка border элемента располагаются за пределами указанной ширины и высоты. Значение по умолчанию.
border-box	Любые padding или border, заданные для элемента, размечаются и отрисовываются внутри указанных значений ширины и высоты. Ширина и высота содержимого вычисляются путем вычитания ширины границ и полей соответствующих сторон из указанных свойств ширины и высоты. Значение auto свойств width и height не зависит от свойства box-sizing и всегда устанавливает размер блока с содержимым. Сумма padding и border не должна превышать заданные значения width и height, в противном случае размер области содержимого будет равен нулю.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Некоторые html-элементы, например, `<button>`, по умолчанию имеют `box-sizing: border-box`.

```
box-sizing: content-box;  
box-sizing: border-box;  
box-sizing: inherit;  
box-sizing: initial;
```

## CSS-позиционирование

CSS рассматривает макет html-документа как дерево элементов. Уникальный элемент, у которого нет родительского элемента, называется **корневым** элементом. Модуль CSS-позиционирования описывает, как любой из элементов может быть размещен независимо от порядка документа (т.е. извлечен из «потока»).

В CSS2 каждый элемент в дереве документа генерирует ноль или более блоков в соответствии с блочной моделью. Модуль CSS3 дополняет и расширяет схему позиционирования. Расположение этих блоков регулируется:

- размерами и типом элемента,
- схемой позиционирования (нормальный поток, обтекание и абсолютное позиционирование),
- отношениями между элементами в дереве документа,
- внешней информацией (например, размер области просмотра, внутренними размерами изображений и т.д.).

## Схемы позиционирования

В CSS блок элемента может быть расположен в соответствии с тремя схемами позиционирования.

### Нормальный поток

Нормальный поток включает блочный контекст форматирования (элементы с `display block`, `list-item` или `table`), строчный (встроенный) контекст форматирования (элементы с `display inline`, `inline-block` или `inline-table`), и относительное и «липкое» позиционирование элементов уровня блока и строки.

### Обтекание

В обтекающей модели блок удаляется из нормального потока и позиционируется влево или вправо. Содержимое обтекает правую сторону элемента с `float: left` и левую сторону элемента с `float: right`.

### Абсолютное позиционирование

В модели абсолютного позиционирования блок полностью удаляется из нормального потока и ему присваивается позиция относительно содержащего блока. Абсолютное позиционирование реализуется с помощью значений `position: absolute`; и `position: fixed`;

Элементом «вне потока» может быть плавающий, абсолютно позиционированный или корневой элемент.

## Содержащий блок

Положение и размер блока(ов) элемента иногда вычисляются относительно некоторого прямоугольника, называемого содержащим блоком элемента (`containing block`). В общих словах, содержащий блок – это блок, который содержит другой элемент. В случае нормального потока корневой элемент `html` является содержащим блоком для элемента `body`, который, в свою очередь, является содержащим блоком для всех его дочерних элементов и так далее. В случае позиционирования содержащий блок полностью зависит от типа

позиционирования. Содержащий блок элемента определяется следующим образом:

- Содержащий блок, в котором находится корневой элемент, представляет собой прямоугольник – так называемый начальный содержащий блок.
- Для некорневого элемента с `position: static;` или `position: relative;` содержащий блок формируется краем области содержимого ближайшего родительского блока уровня блока, ячейки таблицы или уровня строки.
- Содержащим блоком элемента с `position: fixed;` является окно просмотра.
- Для некорневого элемента с `position: absolute;` содержащим блоком устанавливается ближайший родительский элемент со значением `position: absolute/relative/fixed` следующим образом:
  - если предок – элемент уровня блока, содержащим блоком будет область содержимого плюс поля элемента `padding;`
  - если предок – элемент уровня строки, содержащим блоком будет область содержимого;
  - если предков нет, то содержащий блок элемента определяется как начальный содержащий блок.
- Для «липкого» блока содержащим блоком является ближайший предок с прокруткой или окно просмотра, в противном случае.

### Выбор схемы позиционирования: свойство `position`

Свойство `position` определяет, какой из алгоритмов позиционирования используется для вычисления положения блока.

Свойство не наследуется.

Значение	<code>position</code>
<code>static</code>	Блок располагается в соответствии с нормальным потоком. Свойства <code>top</code> , <code>right</code> , <code>bottom</code> и <code>left</code> не применяются. Значение по умолчанию.
<code>relative</code>	<p>Положение блока рассчитывается в соответствии с нормальным потоком. Затем блок смещается относительно его нормального положения и во всех случаях, включая элементы таблицы, не влияет на положение любых следующих блоков. Тем не менее, такое смещение может привести к перекрытию блоков, а также к появлению полосы прокрутки в случае переполнения.</p> <p>Относительно позиционированный блок сохраняет свои размеры, включая разрывы строк и пространство, первоначально зарезервированное для него.</p> <p>Относительно позиционированный блок создает новый содержащий блок для абсолютно позиционированных потомков.</p> <p>Влияние <code>position: relative;</code> на элементы таблицы определяется следующим образом:</p> <p>Элементы с <code>table-row-group</code>, <code>table-header-group</code>, <code>table-footer-group</code> и <code>table-row</code> смещаются относительно их обычной позиции в таблице. Если ячейки таблицы занимают несколько строк, смещаются только ячейки начальной строки.</p> <p><code>table-column-group</code>, <code>table-column</code> не смещает соответствующий столбец и не оказывает визуального влияния.</p> <p><code>table-caption</code> и <code>table-cell</code> смещаются относительно своего нормального положения в таблице. Если ячейка таблицы охватывает несколько столбцов или строк, то она смещается целиком.</p>



absolute	<p>Положение блока (и, возможно, размер) задается с помощью свойств <code>top</code>, <code>right</code>, <code>bottom</code> и <code>left</code>. Эти свойства определяют явное смещение относительно его содержащего блока. Абсолютно позиционированные блоки полностью удаляются из нормального потока, не влияя на расположение сестринских элементов.</p> <p>Отступы <code>margin</code> абсолютно позиционированных блоков не схлопываются.</p> <p>Абсолютно позиционированный блок создает новый содержащий блок для дочерних элементов нормального потока и потомков с <code>position: absolute</code>;</p> <p>Содержимое абсолютно позиционированного элемента не может обтекать другие блоки. Абсолютно позиционированный блок могут скрывать содержимое другого блока (или сами могут быть скрыты), в зависимости от значения <code>z-index</code> перекрывающихся блоков.</p>
sticky	<p>Положение блока рассчитывается в соответствии с нормальным потоком. Затем блок смещается относительно своего ближайшего предка с прокруткой или окна просмотра, если ни у одного из предков нет прокрутки.</p> <p>«Липкий» блок может перекрывать другие блоки, а также создавать полосы прокрутки в случае переполнения.</p> <p>«Липкий» блок сохраняет свои размеры, включая разрывы строк и пространство, первоначально зарезервированное для него.</p> <p>«Липкий» блок создает новый содержащий блок для абсолютно и относительно позиционированных потомков.</p>
fixed	<p>Фиксированное позиционирование аналогично абсолютному позиционированию, с отличием в том, что для содержащим блоком устанавливается окно просмотра. Такой блок полностью удаляется из потока документа и не имеет позиции относительно какой-либо части документа. Фиксированные блоки не перемещаются при прокрутке документа. В этом отношении они похожи на фиксированные фоновые изображения.</p> <p>При печати фиксированные блоки повторяются на каждой странице, содержащим блоком для них устанавливается область страницы. Блоки с фиксированным положением, которые больше области страницы, обрезаются.</p>
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
position: static;
position: relative;
position: absolute;
position: sticky;
position: fixed;
position: initial;
position: inherit;
```

```
Заголовок {position: absolute; top: 10px;}
```

```
Заголовок {position: static}
```

Заголовок с `position static` сохраняет своё место в потоке документа

```
Заголовок {position: relative; top: 10px;}
```

Заголовок с `position relative` смещается относительно своего текущего положения в макете

Заголовок с `position absolute` смещается относительно верхнего левого угла браузера. При этом его ширина становится равной ширине содержимого

## Смещение блока: свойства `top`, `right`, `bottom`, `left`

Элемент считается позиционированным, если свойство `position` имеет значение, отличное от `static`. Позиционированные элементы генерируют позиционированные блоки и могут быть расположены в соответствии со следующими четырьмя физическими свойствами:

Значение	<code>top</code>
<code>auto</code>	Влияние значения зависит от типа элемента. Значение по умолчанию.
длина	Смещение на фиксированном расстоянии от указанного края. Отрицательные значения допускаются.
%	Процентные значения вычисляются относительно высоты содержащего блока. Для «липкого» блока – относительно высоты корневого элемента. Отрицательные значения допускаются.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
top: 10px;  
top: 2em;  
top: 50%;  
top: auto;  
top: inherit;  
top: initial;
```

Свойство `top` задает расстояние, на которое верхний край абсолютно позиционированного блока (с учетом его `margin`) смещается ниже верхнего края содержащего блока. Для относительно позиционированных блоков определяет смещение относительно верхнего края самого блока (то есть блоку задается позиция в нормальном потоке, а затем смещение от этой позиции в соответствии с этим свойством).

Значение	<code>right</code>
<code>auto</code>	Влияние значения зависит от типа элемента. Значение по умолчанию.
длина	Смещение на фиксированном расстоянии от указанного края. Отрицательные значения допускаются.
%	Процентные значения вычисляются относительно ширины содержащего блока. Для «липкого» блока – относительно ширины корневого элемента. Отрицательные значения допускаются.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
right: -10px;  
right: .5em;  
right: -10%;  
right: auto;  
right: inherit;  
right: initial;
```

Свойство `right` указывает расстояние, на которое правый край абсолютно позиционированного блока (с учетом его `margin`) смещен влево от правого края содержащего блока. Для относительно позиционированных блоков определяет смещение относительно правого края самого блока.

Значение	<code>bottom</code>
<code>auto</code>	Влияние значения зависит от типа элемента. Значение по умолчанию.
длина	Смещение на фиксированном расстоянии от указанного края. Отрицательные значения допускаются.
%	Процентные значения вычисляются относительно высоты содержащего блока. Для «липкого» блока – относительно высоты корневого элемента. Отрицательные значения допускаются.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
bottom: 50px;  
bottom: -3em;  
bottom: -50%;  
bottom: auto;  
bottom: inherit;  
bottom: initial;
```

Свойство `bottom` указывает расстояние, на которое нижний край блока смещен вверх относительно нижнего края содержащего блока. Для относительно позиционированных блоков определяет смещение относительно нижнего края самого блока.

Значение	<code>left</code>
<code>auto</code>	Влияние значения зависит от типа элемента. Значение по умолчанию.
длина	Смещение на фиксированном расстоянии от указанного края. Отрицательные значения допускаются.
%	Процентные значения вычисляются относительно ширины содержащего блока. Для «липкого» блока – относительно ширины корневого элемента. Отрицательные значения допускаются.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
left: 50px;  
left: 10em;  
left: 20%;  
left: auto;  
left: inherit;  
left: initial;
```

Свойство `left` указывает расстояние, на которое левый край смещен вправо от левого края содержащего блока. Для относительно позиционированных блоков определяет смещение относительно левого края самого блока.

Положительные значения смещают элемент внутрь содержащего блока, а отрицательные – за его пределы.

### Обтекание: свойство float

Обтекание позволяет блокам смещаться влево или вправо на текущей строке. «Плавающий блок» смещается влево или вправо до тех пор, пока его внешний край не коснется края содержащего блока или внешнего края другого плавающего блока. Если имеется линейный блок, внешняя верхняя часть плавающего блока выравнивается с верхней частью текущего линейного блока.

При использовании свойства `float` для элементов рекомендуется задавать ширину. Тем самым браузер создаст место для другого содержимого. Если для плавающего элемента недостаточно места по горизонтали, он будет смещаться вниз до тех пор, пока не уместится. При этом остальные элементы уровня блока будут его игнорировать, а элементы уровня строки будут смещаться вправо или влево, освобождая для него пространство и обтекая его.

Правила, регулирующие поведение плавающих боков, описываются свойством `float`.

Свойство не наследуется.

Значение	<code>float</code>
<code>none</code>	Отсутствие обтекания. Значение по умолчанию.
<code>left</code>	Элемент перемещается влево, содержимое обтекает плавающий блок по правому краю.
<code>right</code>	Элемент перемещается вправо, содержимое обтекает плавающий блок по левому краю.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

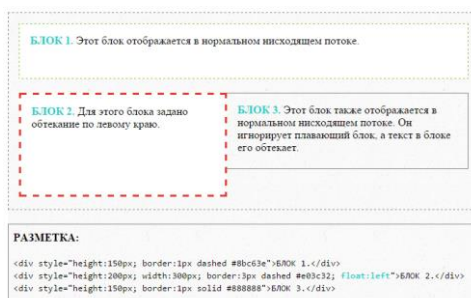
```
float: left;  
float: right;  
float: none;  
float: inherit;
```

Плавающий блок принимает размеры своего содержимого с учетом внутренних отступов и рамок. Верхние и нижние отступы `margin` плавающих элементов не схлопываются.

Плавающие элементы могут использовать отрицательные отступы `margin`, чтобы перемещаться за пределы области содержимого их родительского элемента.

Свойство автоматически изменяет вычисляемое (отображаемое в браузере) значение свойства `display` на `display: block` для следующих значений: `inline`, `inline-block`, `table-row`, `table-row-group`, `table-column`, `table-column-group`, `table-cell`, `table-caption`, `table-header-group`, `table-footer-group`. Значение `inline-table` меняет на `table`.

Свойство не оказывает влияние на элементы с `display: flex` и `display: inline-flex`. Не применяется к абсолютно позиционированным элементам.



## Управление потоком рядом с плавающими элементами: свойство clear

Свойство `clear` указывает, какие стороны блока/блоков элемента не должны прилегать к плавающим блокам, находящимся выше в исходном документе. В CSS2 и CSS2.1 свойство применяется только к неплавающим элементам уровня блока.

Свойство не наследуется.

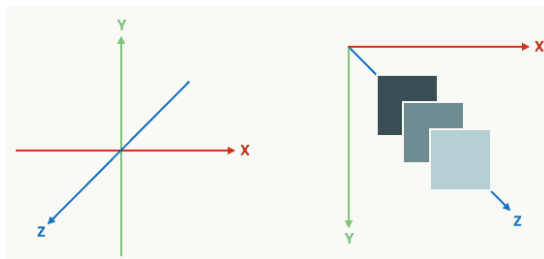
Значение	<code>clear</code>
<code>none</code>	Означает отсутствие ограничений на положение элемента относительно плавающих блоков. Значение по умолчанию.
<code>left</code>	Смещает элемент вниз относительно нижнего края любого плавающего слева элемента, находящегося выше в исходном документе.
<code>right</code>	Смещает элемент вниз относительно нижнего края любого плавающего справа элемента, находящегося выше в исходном документе.
<code>both</code>	Смещает элемент вниз относительно нижнего края любого плавающего слева и справа элемента, находящегося выше в исходном документе.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
clear: none;  
clear: left;  
clear: right;  
clear: both;  
clear: inherit;
```

Для предотвращения отображения фона или границ под плавающими элементами используется правило `{overflow: hidden;}`.

## Определение контекста наложения: свойство z-index

В CSS каждый блок имеет позицию в трех измерениях. В дополнение к горизонтальному и вертикальному положению, блоки выкладываются вдоль оси Z друг над другом. Положение вдоль оси Z особенно важно, когда блоки визуально накладываются друг на друга.



Порядок, в котором дерево документа отрисовывается на экране, описывается с помощью **контекста наложения**. Каждый блок принадлежит одному контексту наложения. Каждый блок в данном контексте наложения имеет целочисленный уровень, который является его положением на оси Z относительно других блоков в том же контексте наложения.

Блоки с более высокими уровнями всегда отображаются перед блоками с более низкими уровнями, а блоки с одинаковым уровнем располагаются снизу вверх в соответствии с порядком следования элементов в исходном документе. Блок элемента имеет ту же позицию, что и блок его родителя, если только ему не присвоен другой уровень свойством `z-index`.

Свойство `z-index` позволяет изменить порядок наложения позиционированных элементов в случае, когда они накладываются друг на друга.

Свойство не наследуется.

Значение	z-index
auto	Вычисляется в 0. Если для блока задано <code>position: fixed;</code> или это корневой элемент, значение <code>auto</code> также устанавливает новый контекст наложения. Значение по умолчанию.
целое число	Определяет положение блока в текущем контексте наложения. Также устанавливает новый локальный контекст наложения. Можно использовать любое целое число, включая отрицательные числа. Отрицательные значения помещают элемент вглубь экрана.
inherit	Наследует значение свойства от родительского элемента.
initial	Устанавливает значение свойства в значение по умолчанию.

```
z-index: auto;  
z-index: 0;  
z-index: 5;  
z-index: 999;  
z-index: -1;  
z-index: inherit;  
z-index: initial;
```

### Контекст наложения

Если для элементов свойства `z-index` и `position` не заданы явно, контекст наложения равен порядку их расположения в исходном коде и браузер отображает элементы на странице в следующем порядке:

- Корневой элемент `<html>`, который содержит все элементы веб-страницы.
- Блочные элементы, неплавающие и непозиционированные.
- Плавающие `float` непозиционированные элементы в порядке их расположения в исходном коде.
- Строковые непозиционированные элементы (текст, изображения).
- Позиционированные `position` элементы в порядке их следования в исходном коде. Последний из них будет расположен на переднем плане.

Свойство `z-index` создает новый контекст наложения. Оно позволяет изменить порядок наложения позиционированных элементов. Элементы будут отображаться на странице в следующем порядке (если для них не заданы свойства, влияющие на контекст наложения – `opacity`, `filter`, `transform`):

- Корневой элемент `<html>`, который содержит все элементы веб-страницы.
- Позиционированные элементы с отрицательным значением `z-index`.
- Блочные элементы, неплавающие и непозиционированные.
- Плавающие `float` непозиционированные элементы в порядке их расположения в исходном коде.
- Строковые непозиционированные элементы (текст, изображения).
- Позиционированные элементы со значениями `z-index: 0;` и `z-index: auto;`.

### 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. CSS построен на [Unicode](#).

### 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.

Значение	<code>text-transform</code>
<code>none</code>	Значение по умолчанию, означает отсутствие эффектов.
<code>capitalize</code>	Изменяет написание первой буквы каждого слова в элементе, делая ее прописной.
<code>uppercase</code>	Выводит все слова в элементе прописными буквами.
<code>lowercase</code>	Выводит все слова в элементе строчными буквами.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
text-transform: none;
text-transform: capitalize;
text-transform: uppercase;
text-transform: lowercase;
text-transform: inherit;
text-transform: initial;
```

### Обработка пробелов и переносы строк: свойство `white-space`

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

Свойство наследуется.

Значение	<code>white-space</code>
<code>normal</code>	Значение по умолчанию. Между словами вставляется только по одному пробелу, дополнительные пробелы отбрасываются. Текст переносится только в случае необходимости.
<code>nowrap</code>	Запрещает переносы строк, за исключением применения <code>&lt;br&gt;</code> .
<code>pre</code>	Пробелы в тексте не игнорируются, браузер отображает дополнительные пробелы и переносы строк.
<code>pre-wrap</code>	Сохраняет пробелы в тексте, делая разрывы строк там, где это необходимо.
<code>pre-line</code>	Удаляет лишние пробелы, за исключением случаев <code>&lt;br&gt;</code> .
<code>break-spaces</code>	Поведение идентично <code>pre-wrap</code> , 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.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.



```
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;
```

### Настройка табуляции: свойство tab-size

Свойство `tab-size` используется для изменения величины отступа, получаемого с помощью клавиши TAB. Значения свойства игнорируются, когда установлено одно из трех 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, поддерживаются только в Chrome 42+.

Свойство наследуется.

Значение	tab-size
целое число	Любое целое положительное число. По умолчанию табуляция делает отступ, равный восьми пробелам.
длина	Значение отступа, указываемое в единицах длины, например, <code>px</code> .
pre	Пробелы в тексте не игнорируются, браузер отображает дополнительные пробелы и переносы строк.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
tab-size: 0;
tab-size: 10px;
tab-size: inherit;
tab-size: initial;
```

### 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.

#### 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.

Значение	word-break
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.

```
word-break: normal;
word-break: break-all;
word-break: keep-all;
word-break: inherit;
word-break: initial;
```

### 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.

Значение	line-break
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.

```
line-break: auto;
line-break: loose;
line-break: normal;
line-break: strict;
line-break: inherit;
line-break: initial;
```

### 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.

Значение	hyphens
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.

```
hyphens: none;
hyphens: manual;
hyphens: auto;
hyphens: inherit;
hyphens: initial;
```

### 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.

Значение	overflow-wrap, word-wrap
normal	Неразрывные строки могут разрываться только в разрешенных точках разрыва. Значение по умолчанию.
break-word	Перенос строк добавляется автоматически, чтобы слово поместилось в заданную ширину блока.
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.

```
overflow-wrap: normal;
overflow-wrap: break-word;
overflow-wrap: anywhere;
overflow-wrap: inherit;
overflow-wrap: initial;
```

### Выравнивание и выключение строк

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

## 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.

Значение	<code>text-align</code>
<code>start</code>	Содержимое на уровне строки выравнивается по начальному краю линейного блока. Значение по умолчанию.
<code>end</code>	Содержимое на уровне строки выравнивается по конечному краю линейного блока.
<code>right</code>	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.
<code>center</code>	Содержимое на уровне строки центрируется внутри линейного блока.
<code>justify</code>	Текст выравнивается по ширине линейного блока, чтобы точно заполнить поле строки, прижимаясь к левому и правому краям родительского элемента. Если иное не указано в <code>text-align-last</code> , 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.
<code>justify-all</code>	Устанавливает <code>text-align-all</code> и <code>text-align-last</code> в <code>justify</code> , также выравнивая последнюю строку.
<code>match-parent</code>	Значение ведет себя так же, как <code>inherit</code> за исключением того, что унаследованное значение <code>start</code> или <code>end</code> интерпретируется относительно значения <code>direction</code> (или исходного содержащего блока, если нет родителя) и приводит к вычисленному значению <code>left</code> или <code>right</code> .
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
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;
```

## 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.

```
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;
```

### 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.

```
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;
```

## Промежутки

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

### 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.

Значение	<code>word-spacing</code>
<code>normal</code>	Дополнительный интервал не применяется. Вычисляет в 0. Значение по умолчанию.
длина	Задаёт дополнительный интервал в дополнение к внутреннему интервалу между словами, определённому шрифтом. Значения могут быть отрицательными, но могут быть ограничения, зависящие от реализации.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
word-spacing: normal;
word-spacing: 1px;
word-spacing: 0.2em;
word-spacing: 1rem;
word-spacing: inherit;
word-spacing: initial;
```

## Трекинг: свойство letter-spacing

Свойство `letter-spacing` определяет дополнительный интервал, или трекинг, между смежными типографскими символами. Межбуквенный интервал является дополнением к кернингу 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.

Значение	letter-spacing
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.

```
letter-spacing: normal;  
letter-spacing: 0.1em;  
letter-spacing: 2px;  
letter-spacing: inherit;  
letter-spacing: initial;
```

## Отступ первой строки: свойство text-indent

Свойство `text-indent` задает отступ, применяемый к строкам встроеного 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.

Значение	text-indent
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.

```
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-шрифты

**Шрифт** в CSS – это ресурс, содержащий визуальное представление символов. На самом простом уровне он содержит информацию, которая

сопоставляет коды символов с фигурами (называемые глифами), представляющие эти символы.

Шрифты, использующие общий стиль дизайна, обычно группируются в семейства шрифтов, классифицируемые набором стандартных свойств шрифта. Внутри семейства форма, отображаемая для данного символа, может варьироваться в зависимости от толщины обводки, наклона или относительной ширины.

Ресурсы шрифтов могут быть установлены локально на устройстве, в котором работает браузер. Для локальных ресурсов шрифта описательная информация может быть получена непосредственно из ресурса шрифта (например, из файла `arial.woff`). Для загружаемых ресурсов шрифтов, также называемых веб-шрифтами, описательная информация включена со ссылкой на ресурс шрифта (например, для шрифта [Poiret One](#)).

Используя различные шрифты для заголовков, абзацев и других элементов, можно задавать определенный стиль письменных сообщений, передавая желаемые эмоции и настроение. Окунаясь в многообразие шрифтов, не забывайте, что текст основного содержимого веб-страницы должен быть в первую очередь читабельным.

Не рекомендуется использовать более двух шрифтов на странице, а желаемого контраста можно достигнуть за счет комбинирования шрифтов разной толщины, размера, начертания или же при помощи цвета.

### Семейство шрифтов: свойство `font-family`

Свойство `font-family` используется для выбора начертания шрифта. Поскольку невозможно предсказать, установлен тот или иной шрифт на компьютере посетителя вашего сайта, рекомендуется прописывать все возможные варианты однотипных шрифтов. В таком случае браузер будет проверять их наличие, последовательно перебирая предложенные варианты.

Если в названии шрифта имеются пробелы или символы (например, #, \$, %), то оно заключается в кавычки. Это делается для того, чтобы браузер мог понять, где начинается и заканчивается название шрифта.

Свойство наследуется.

Значение	<code>font-family</code>
family-name	Название (имя) семейства шрифтов, например, <code>Times</code> , <code>Courier</code> , <code>Arial</code> . Рекомендуется указывать вместе с базовым семейством.
generic-family	Базовое семейство. CSS определяет пять базовых семейств шрифтов: Шрифты с засечками – <code>Serif</code> ( <code>Times New Roman</code> , <code>Times</code> , <code>Garamond</code> , <code>Georgia</code> ). Рубленые шрифты – <code>Sans-serif</code> ( <code>Helvetica</code> , <code>Geneva</code> , <code>Arial</code> , <code>Verdana</code> , <code>Trebuchet</code> , <code>Univers</code> ). Моноширинные шрифты – <code>Monospace</code> ( <code>Courier</code> , <code>Courier New</code> , <code>Andale Mono</code> ). Рукописные шрифты – <code>Cursive</code> ( <code>Comic Sans</code> , <code>Gabriola</code> , <code>Monotype Corsiva</code> , <code>Author</code> , <code>Zapf Chancery</code> ). Аллегорические шрифты ( <code>Western</code> , <code>Woodblock</code> , <code>Klingon</code> ).
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.



```
font-family: "Times New Roman", Georgia, Serif;
font-family: serif;
font-family: sans-serif;
font-family: monospace;
font-family: cursive;
font-family: fantasy;
font-family: system-ui;
font-family: inherit;
font-family: initial;
```

## Насыщенность шрифта: свойство font-weight

Свойство `font-weight` задает насыщенность шрифта.

Свойство наследуется.

Значение	font-weight
normal	Значение по умолчанию, устанавливает нормальную насыщенность шрифта. Эквивалентно значению насыщенности, равной 400.
bold	Делает шрифт текста полужирным. Эквивалентно значению насыщенности, равной 700.
bolder	Насыщенность шрифта будет больше, чем у предка.
lighter	Насыщенность шрифта будет меньше, чем у предка.
100, 200, 300, 400, 500, 600, 700, 800, 900	Значение 100 соответствует самому легкому варианту начертания шрифта, а 900 – самому плотному. При этом, эти числа не определяют конкретной плотности, т.е. 100, 200, 300 и 400 могут соответствовать одному и тому же варианту слабой насыщенности начертания шрифта; 500 и 600 – средней насыщенности, а 700, 800 и 900 могут выводить одинаковое очень насыщенное начертание. Распределение плотности так же зависит от количества уровней насыщенности, определенных в конкретном семействе шрифтов.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
font-weight: normal;
font-weight: bold;
font-weight: lighter;
font-weight: bolder;
font-weight: 100;
font-weight: 200;
font-weight: 300;
font-weight: 400;
font-weight: 500;
font-weight: 600;
font-weight: 700;
font-weight: 800;
font-weight: 900;
font-weight: inherit;
font-weight: initial;
```



## Ширина шрифта: свойство font-stretch

Свойство `font-stretch` позволяет выбрать нормальное, сжатое или расширенное начертание символа из семейства шрифтов. Свойство не работает на любом шрифте, а только на шрифтах, для которых разработаны различными начертания, соответствующими определенным размерам.

Свойство наследуется.

Абсолютные значения ключевых слов имеют следующий порядок, от самого узкого до самого широкого:

Значение	font-stretch
ultra-condensed	Указывает на наиболее сжатый шрифт.
extra-condensed	Указывает на второй по сжатости шрифт.
condensed	Указывает на сжатый шрифт.
semi-condensed	Указывает на немного сжатый шрифт.
normal	Значение по умолчанию.
semi-expanded	Слегка расширенный шрифт.
expanded	Расширенный шрифт.
extra-expanded	Второй по расширенности шрифт.
ultra-expanded	Максимально расширенный шрифт.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
font-stretch: ultra-condensed;  
font-stretch: extra-condensed;  
font-stretch: condensed;  
font-stretch: semi-condensed;  
font-stretch: normal;  
font-stretch: semi-expanded;  
font-stretch: expanded;  
font-stretch: extra-expanded;  
font-stretch: ultra-expanded;  
font-stretch: inherit;  
font-stretch: initial;
```

Когда не существует глифа для заданной ширины, значения **normal** или **condensed** отображаются для более узкого начертания символа, в противном случае отображается более широкое начертание. И наоборот, расширенные значения используют широкое начертание, в противном случае – узкое начертание. На рисунке ниже показано, как девять параметров свойства влияют на выбор шрифта для семейства шрифтов, содержащего различные ширины, серый цвет указывает ширину, для которой не существует начертания, поэтому подставляется другая ширина:



### Начертание шрифта: свойство font-style

Свойство **font-style** позволяет выбрать стиль начертания для шрифта. При этом разница между курсивом и наклонным начертанием заключается в том, что курсив вносит небольшие изменения в структуру каждого символа, в то время как наклонное начертание представляет собой наклонную версию прямого шрифта.

Свойство наследуется.

Значение	font-style
normal	Значение по умолчанию, устанавливает для текста обычное начертание шрифта.
italic	Выделяет текст курсивом.
oblique	Устанавливает наклонное начертание шрифта.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
font-style: normal;  
font-style: italic;  
font-style: oblique;  
font-style: inherit;  
font-style: initial;
```

a a a N N N

### Размер шрифта: свойство font-size

Свойство font-size указывает желаемую высоту глифов из шрифта.

Свойство наследуется.

Значение	font-size
absolute-size	xx-small, x-small, small, medium, large, x-large, xx-large. В качестве стандартного размера принимается medium. В CSS1 предложенный коэффициент масштабирования между соседними индексами составлял 1.5, что для пользователя оказалось слишком большим. В CSS2 предложенный коэффициент масштабирования для экрана компьютера между смежными индексами составлял 1.2, что все еще создавало проблемы для небольших размеров. Новый коэффициент масштабирования варьируется между каждым индексом, чтобы обеспечить лучшую читаемость.
relative-size	smaller, larger. Относительные размеры обуславливают изменение размера шрифта элемента относительно родителя. При этом размер шрифта может выйти за рамки размеров, предполагаемых для xx-small и xx-large.
длина	Размер шрифта устанавливается с помощью положительных значений единиц длины, например, px, em, как целых, так и дробных.
%	Относительное значение, вычисляется на основании любого размера, унаследованного от родительского элемента. Обеспечивает более точную настройку вычисляемого размера шрифта. Задание размеров шрифта с помощью em эквивалентно процентному значению.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
font-size: xx-small;
font-size: x-small;
font-size: small;
font-size: medium;
font-size: large;
font-size: x-large;
font-size: xx-large;
font-size: smaller;
font-size: larger;
font-size: 14px;
font-size: 0.8em;
font-size: 80%;
font-size: inherit;
font-size: initial;
```

Значения absolute-size							
xx-small	x-small	small	medium	large	x-large	xx-large	
Коэффициент масштабирования							
3/5	3/4	8/9	1	6/5	3/2	2/1	3/1
HTML заголовки							
h6		h5	h4	h3	h2	h1	
HTML размер шрифта							
1		2	3	4	5	6	7

### Сокращенная запись свойств шрифта: свойство font

Свойство **font** за исключением описанного ниже, является сокращенным свойством для установки **font-style**, **font-variant**, **font-weight**, **font-stretch**, **font-size/line-height**, **font-family**. Также могут быть включены значения для свойства **font-variant**, которые поддерживаются CSS 2.1 – **normal** или **small-caps**.

Все подсвойства свойства **font** сначала сбрасываются на свои начальные значения, включая перечисленные выше, плюс **font-size-adjust**, **font-kerning**, все подсвойства **font-variant** и настройки шрифтов, за исключением **font-synthesis**. Затем этим свойствам присваиваются те значения, которые указаны в свойстве **font**. Для свойства **font-size-adjust** невозможно установить значение, отличное от его начального значения, поэтому следует использовать вместо этого индивидуальное свойство. Если явное значение какого-либо свойства не нужно, то оно опускается.

Свойство наследуется.

```
font: 12pt/14pt sans-serif;
font: 80% sans-serif;
font: x-large/110% "new century schoolbook", serif;
font: bold italic large Palatino, serif;
font: normal small-caps 120%/120% fantasy;
font: condensed oblique 12pt "Helvetica Neue", serif;
```

Следующие значения относятся к системным шрифтам:

- **caption** – шрифт, используемый для элементов управления с субтитрами (например, кнопок, раскрывающихся списков и т.д.).
- **icon** – шрифт, используемый для обозначения значков.
- **menu** – шрифт, используемый в меню (например, раскрывающиеся меню и списки меню).
- **message-box** – шрифт, используемый в диалоговых окнах.
- **small-caption** – шрифт, используемый для маркировки подписи элементов управления.
- **status-bar** – шрифт, используемый в строке состояния окна.

Системные шрифты могут быть установлены только целиком; то есть семейство шрифтов, размер, вес, стиль и т.д. задаются одновременно. Эти значения затем могут быть изменены индивидуально, если это необходимо. Ключевые слова, используемые для системных шрифтов, перечисленных выше, обрабатываются как ключевые слова только в том случае, если они находятся в начальной позиции, в других позициях эта же строка обрабатывается как часть имени семейства шрифтов. Системные шрифты могут быть указаны только с этим свойством, но не с самим `font-family`.

```
font: menu; /* используются настройки шрифта для системных меню */
font: large menu; /* используется семейство шрифтов под названием "menu" */
```

## CSS-ссылки

**CSS-ссылки** содержат свойства, которые отвечают за внешний вид гипертекстовых ссылок HTML-документа. Ссылки представляют собой основной способ навигации по сайту, поэтому применение CSS-стилей для оформления улучшит их визуальное восприятие.

Основной способ оформления ссылок заключается в стилизации подчеркивания ссылки и изменении цвета текста ссылки. Также можно изменить внешний вид курсора с помощью свойства `cursor`.

## Псевдоклассы состояний гипертекстовых ссылок

Большинство браузеров выделяют четыре основных состояния гиперссылок, каждому из которых соответствует свой псевдокласс селектора:

- Непосещенная – `a:link`.
- Посещенная – по которой уже выполнялся переход – `a:visited`.
- Не нажатая – над которой находится указатель мыши – `a:hover`.
- Нажатая – которая удерживается мышью – `a:active`.

Используя псевдоклассы для форматирования каждого состояния ссылок, можно дать пользователям подсказки, по каким ссылкам он уже переходил, а по каким – еще нет, например:

```
a:link {
  color: #4970DD;
  border-bottom: 1px dashed;
}
a:visited {
  color: #EF7D55;
}
a:hover {
  color: #154088;
  border-bottom: .07em solid;
}
a:active {
  color: #4970DD;
  border-bottom: 1px dashed;
}
```

Форматировать ссылки нужно в указанной последовательности, в противном случае состояние стилей перестанет работать (в силу механизма каскадности).

## Выборка отдельных ссылок

Для стилизации отдельных ссылок нужно задать им стилевой класс, после чего можно будет менять внешний вид выбранных ссылок:

```
<a href="http://bsuir.by" class="global">какой-то текст</a>
```

## Подчеркивание ссылок

Удаление подчеркивания:

```
a {
  text-decoration: none;
}
```

Добавление подчеркивания только при наведении на ссылку:

```
a {
  text-decoration: none;
}
a:hover {
  text-decoration: underline;
}
```

Внешний вид нижней границы ссылки:

```
a {
  text-decoration: none;
  border-bottom: 2px dashed DarkOrchid;
  padding-bottom: 3px;
}
```

## Изображения для ссылок

Добавить изображение для ссылки можно с помощью CSS-свойства `background-image`. Так как элемент `<a>` является строчным `a {display: inline;}`, то предварительно его нужно преобразовать в блочный элемент `a {display: block;}`.

Чтобы вставить изображение или иконку перед ссылкой, необходимо добавить отступ с помощью свойства `padding-left`. Этот прием может пригодиться в случае, когда на странице есть ссылки для загрузки каких-либо документов различных форматов, и вы можете добавить значок-изображение типа файла для большей наглядности.

Если нужно, чтобы значок автоматически добавился ко всем ссылкам, содержащим документы одного формата, можно воспользоваться следующей конструкцией:

```
a[href$=".pdf"] {
  background-image: url(images/pdf.png);
}
```

Символ `href$` в селекторе атрибута дает браузеру команду найти все атрибуты `href`, заканчивающиеся определенным образом (в данном случае `.pdf`) и добавить к ссылке соответствующий значок.

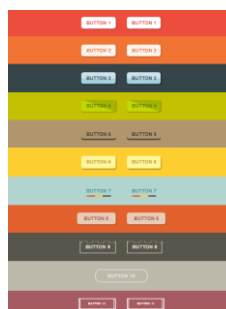
## Использование фонового изображения

Можно преобразовать внешний вид ссылки, добавив в качестве нижней границы фоновое изображение:

```
a {
  text-decoration: none;
  background: url(images/underline.png) repeat-x left bottom;
  padding-bottom: 3px;
}
```

## Ссылки-кнопки

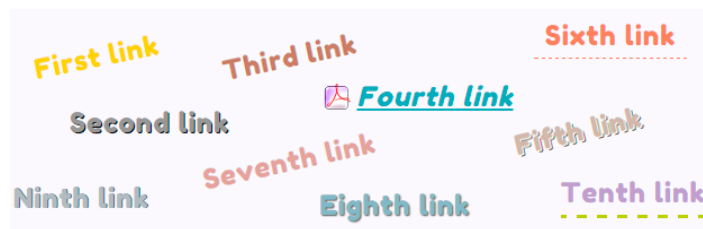
Благодаря свойствам `background-color`, `border` и `padding`, ссылкам можно придать вид прямоугольных кнопок, а, меняя отображение тех или иных свойств ссылок при наведении курсора мыши `a:hover`, добавить интересные эффекты.



[Смотреть пример](#)

## Примеры оформления ссылок

Гипертекстовые ссылки можно оформить различными способами, но основной прием оформления основывается на изменении внешнего вида ссылки при наведении на нее курсором мыши – состояние ссылки `a:hover`.



[Смотреть пример](#)

## CSS-таблицы

Спецификация CSS дает 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.

### Границы таблицы border

Таблица и ячейки внутри нее 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*/
}
```

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

```
th {
  border: 1px solid grey;
}
```

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

```
td {
  border: 1px solid grey;
}
```

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

```
th, td {
  border: 1px solid grey;
}
```

Внешнюю границу таблицы можно выделить, задав ей увеличенную ширину:

```
table {
  border: 3px solid grey;
}
```

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;
}
```



## 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 ее ячеек могут выступать:

- заливка сплошным цветом,
- градиентная заливка,
- фоновое изображение.

## 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.

## 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.

Значение	<code>caption-side</code>
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>
```

Company	Q1	Q2	Q3	Q4
Microsoft	20.3	30.5	23.5	40.3
Google	50.2	40.63	45.23	39.3
Apple	25.4	30.2	33.3	36.7
IBM	20.4	15.6	22.3	29.3

Таблица № 1

```
caption {
  caption-side: bottom;
  text-align: right;
  padding: 10px 0;
  font-size: 14px;
}
```

## Как убрать промежуток между рамками ячеек

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

Свойство наследуется.

Значение	border-collapse
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.

```
table {  
  border-collapse: collapse;  
}
```

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	23.5
Google	50.2	40.63	45.23
Apple	25.4	30.2	33.3
IBM	20.4	15.6	22.3

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	23.5
Google	50.2	40.63	45.23
Apple	25.4	30.2	33.3
IBM	20.4	15.6	22.3

## 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.

Значение	border-spacing
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.

```
table {  
  border-collapse: separate;  
  border-spacing: 10px 20px;  
}  
  
table {  
  border-collapse: separate;  
  border-spacing: 10px;  
}
```

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	23.5
Google	50.2	40.63	45.23
Apple	25.4	30.2	33.3
IBM	20.4	15.6	22.3

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	23.5
Google	50.2	40.63	45.23
Apple	25.4	30.2	33.3
IBM	20.4	15.6	22.3

## 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.

Значение	empty-cells
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 <code>display: none</code> .
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>
```

```
table {
  border: 1px solid #69c;
  border-collapse: separate;
  empty-cells: hide;
}
th, td {
  border: 2px solid #69c;
}
```

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	
Google	50.2	40.63	45.23

### 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.

Значение	table-layout
auto	Значение по умолчанию. Ширина макета таблицы определяется шириной ее содержимого с учетом значений свойств <code>padding-left</code> , <code>padding-right</code> , <code>border-left width</code> плюс одна ширина <code>border-right</code> 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.

```
table {
  table-layout: fixed;
}
```

## Примеры оформления таблиц

### 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`.

### 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.

### «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.

### Горизонтальная зебра

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

### 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.

### 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.

[Смотреть пример](#)

## 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`.

### Тип маркера списка `list-style-type`

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

Свойство наследуется.

Значение	<code>list-style-type</code>
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: а, и, у, е, о, ...
hiragana-iroha	Японская нумерация: и, ро, ха, ни, хо, ...
katakana	Японская нумерация: А, И, У, Е, О, ...
katakana-iroha	Японская нумерация: И, РО, ХА, НИ, ХО, ...
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.

```
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: upper-roman;
list-style-type: upper-latin;
list-style-type: lower-roman;
list-style-type: lower-latin;
list-style-type: lower-greek;
list-style-type: upper-alpha;
list-style-type: lower-alpha;
list-style-type: none;
list-style-type: inherit;
list-style-type: initial;
```

- WordPress
- Joomla
- ModX
- TextPattern
- Drupal

- a. WordPress
- b. Joomla
- c. ModX
- d. TextPattern
- e. Drupal

## Изображения для элементов списка list-style-image

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

Свойство наследуется.

Значение	list-style-image
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.

```
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;
```

- WordPress
- Joomla
- ModX
- TextPattern
- Drupal

## Местоположение маркера списка list-style-position

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

Свойство наследуется.

Значение	list-style-position
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.

```
list-style-position: inside;
list-style-position: outside;
list-style-image: inherit;
list-style-image: initial;
```

• WordPress	a. WordPress
• Joomla	b. Joomla
• ModX	c. ModX
• TextPattern	d. TextPattern
• Drupal	e. Drupal

## 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.

```
ul {
  list-style: url("images/romb.png") inside;
}
```

## Примеры оформления списков



[Смотреть пример](#)

## CSS-фон

Каждый блок html-элемента имеет фоновый слой, который может быть полностью прозрачным (по умолчанию) или заполнен цветом и/или одним или несколькими изображениями. CSS-свойства фона указывают, какой цвет `background-color` и изображения `background-image` использовать, а также их размер, расположение, способ укладки и т.д.

Свойства фона не наследуются, но фон родительского блока будет просвечивать по умолчанию из-за начального значения в `background-color: transparent`.

Фон не отображается у пустых элементов с нулевой высотой. Отрицательные значения свойства `margin` не влияют на фон элемента.

### Базовый цвет: свойство `background-color`

Свойство `background-color` устанавливает цвет фона элемента. Цвет рисуется за фоновыми изображениями. Для блочных элементов цвет фона распространяется на всю ширину и высоту блока элемента, для строчных – только на область их содержимого.

Цвет фона обрезается в соответствии со значением `background-clip` самого нижнего слоя фонового изображения.

Свойство не наследуется.

Значение	<code>background-color</code>
цвет	Значение принимает все форматы цвета свойства <code>color</code> . Значение по умолчанию <code>transparent</code> .
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
background-color: salmon;
background-color: #00ff00;
background-color: rgba(255, 128, 128, 0.5);
background-color: currentColor;
background-color: transparent;
background-color: inherit;
background-color: initial;
```

#### Заголовок

Какой-то текст, а это **СЛОВО** выделено другим цветом.

### Источник изображения: свойство `background-image`

Свойство `background-image` устанавливает фоновое изображение (одно или несколько) элемента. Значение `none` считается слоем изображения, но ничего не рисует. Изображение, которое является пустым (нулевой ширины или нулевой высоты), которое не загружается или не может быть отображено (например, потому что оно не в поддерживаемом формате изображения) также считается слоем, но ничего не рисует.

Семантически важные изображения должны предоставляться в разметке документа, например, с тегом `<img>`.

Свойство не наследуется.

Значение	<code>background-image</code>
изображение	Обозначает 2D-изображение. Это может быть ссылка на URL, нотация <code>image()</code> или запись градиента. Значение по умолчанию <code>none</code> .
inherit	Наследует значение свойства от родительского элемента.



```
background-image: none;
background-image: url(http://site.by/rose.png);
background-image: url(tl.png), url(tr.png);
background-image: linear-gradient(white, gray);
background-image: repeating-radial-gradient(circle closest-side at 20px 30px, red, yellow, green 100%, yellow 150%, red 200%);
background-image: image("sprites.svg#xywh=40,0,20,20");
background-image: inherit;
```

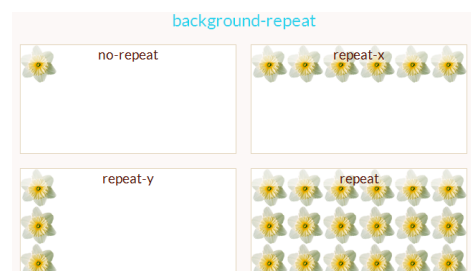
## Укладка изображений: свойство background-repeat

Свойство `background-repeat` определяет, как фоновые изображения укладываются в области фона после того, как для них установлены размеры и позиционирование. Если значение свойства имеет два ключевых слова, первое используется для горизонтального направления, второе – для вертикального.

Свойство не наследуется.

Значение	background-repeat
repeat-x	Изображение повторяется в горизонтальном направлении. Вычисляется в <code>repeat no-repeat</code> .
repeat-y	Изображение повторяется в вертикальном направлении. Вычисляется в <code>no-repeat repeat</code> .
repeat	Изображение повторяется в обоих направлениях так часто, чтобы покрыть область отрисовки фона. Если изображение не помещается, оно обрезается. Вычисляется в <code>repeat repeat</code> . Значение по умолчанию.
space	Изображение повторяется столько раз, сколько оно помещается в области фона, не обрезаясь, изображения расположены на равном расстоянии друг от друга. Первое и последнее изображения касаются краев области. Если область рисования фона больше, чем область позиционирования фона, шаблон повторяется, чтобы заполнить область рисования фона. Если недостаточно места для двух копий изображения, то размещается только одно изображение, а свойство <code>background-position</code> определяет его положение. Вычисляется в <code>space space</code> .
round	Изображение повторяется так часто, чтобы заполнить область фона, масштабируясь и не обрезаясь. Вычисляется в <code>round round</code> .
no-repeat	Изображение размещается один раз и не повторяется. Вычисляется в <code>no-repeat no-repeat</code> .
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
background-repeat: repeat-x;
background-repeat: repeat-y;
background-repeat: repeat;
background-repeat: space;
background-repeat: round;
background-repeat: no-repeat;
background-repeat: repeat space;
background-repeat: repeat repeat;
background-repeat: round space;
background-repeat: no-repeat round;
background-repeat: inherit;
background-repeat: initial;
```



## Фиксация изображения: свойство background-attachment

Свойство `background-attachment` указывает, является ли фоновое изображение фиксированным относительно области просмотра или прокручивается вместе с элементом или его содержимым.

Свойство не наследуется.

Значение	background-attachment
scroll	Фоновое изображение прокручивается вместе с текстом и другим содержимым. Значение по умолчанию.
fixed	Предотвращает перемещение, фиксирует фоновое изображение на заднем плане.
local	Фоновое изображение прокручивается вместе с содержимым элемента.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
background-attachment: scroll;  
background-attachment: fixed;  
background-attachment: local;  
background-attachment: inherit;  
background-attachment: initial;
```

### Позиционирование изображений: свойство background-position

Если для элемента заданы фоновые изображения, свойство `background-position` указывает их начальное положение (после любого изменения размера) в соответствующей области расположения фона.

Свойство не наследуется.

Значение	background-position
%	Горизонтальное смещение вычисляется по формуле ширина области расположения фона – ширина фонового изображения. Вертикальное смещение по формуле высота области расположения фона – высота фонового изображения, где размер изображения – это размер, заданный свойством <code>background-size</code> . Значение по умолчанию <code>0% 0%</code> .
длина	Значение длины дает фиксированную длину в качестве смещения.
left	Вычисляет до <code>0%</code> для горизонтальной позиции, если задано одно или два значения, в противном случае смещение происходит относительно левого края.
center	Вычисляет в <code>left 50%</code> для горизонтального положения, если не указано иное горизонтальное положение, или как <code>top 50%</code> для вертикального положения, если оно задано.
right	Вычисляет в <code>100%</code> для горизонтального положения, если задано одно или два значения, в противном случае смещение происходит относительно правого края.
top	Вычисляет в <code>0%</code> для вертикальной позиции, если задано одно или два значения, в противном случае смещение происходит относительно верхнего края.
bottom	Вычисляет в <code>100%</code> для вертикальной позиции, если задано одно или два значения, в противном случае смещение происходит относительно нижнего края.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

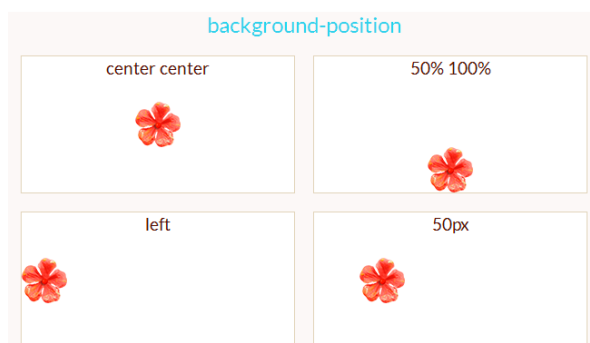
```
background-position: top;
background-position: bottom;
background-position: left;
background-position: right;
background-position: center;
background-position: 25% 75%;
background-position: 0 0;
background-position: 1cm 2cm;
background-position: 10ch 8em;
background-position: 0 0, center;
background-position: bottom 10px right 20px;
background-position: right 3em bottom 10px;
background-position: bottom 10px right;
background-position: top right 10px;
background-position: inherit;
background-position: initial;
```

Если указано только одно значение, второе значение считается `center`. Если заданы два значения в единицах длины или `%`, то первое значения представляет горизонтальную позицию, второе – вертикальную. Значения в единицах длины или `%` представляют смещение верхнего левого угла фонового изображения от верхнего левого угла области расположения фона.

Пара ключевых слов может быть переупорядочена, в то время как комбинация ключевого слова и длины или процента не может. Например, `center left` – допустимое значение, а `50% left` – нет.

Если заданы три или четыре значения в единицах длины или `%`, то перед каждым значением должно стоять ключевое слово, которое указывает, от какого края дается смещение. Если даны три значения, недостающее смещение считается равным нулю.

Положительные значения смещают внутрь от края области расположения фона. Отрицательные значения смещают наружу от края области расположения фона.



### Область рисования: свойство `background-clip`

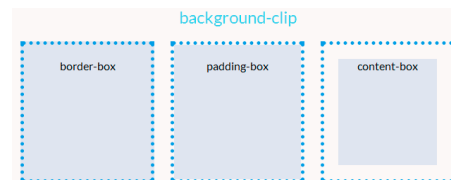
Свойство `background-clip` определяет область рисования фона. Фон всегда рисуется под рамкой элемента, если таковая имеется.

Корневой элемент имеет другую область рисования фона, поэтому свойство `background-clip` на него не влияет.

Свойство не наследуется.

Значение	<code>background-clip</code>
<code>border-box</code>	Фон закрашивает область в пределах рамки элемента. Значение по умолчанию.
<code>padding-box</code>	Фон закрашивает область в пределах внутренних полей элемента.
<code>content-box</code>	Фон закрашивает только область содержимого.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
background-clip: border-box;
background-clip: padding-box;
background-clip: content-box;
background-clip: text;
background-clip: inherit;
background-clip: initial;
```



## Область расположения фона: свойство background-origin

Свойство `background-origin` указывает область расположения фона для элементов, которые выводятся на экране как единый блок (например, не абзацы текста).

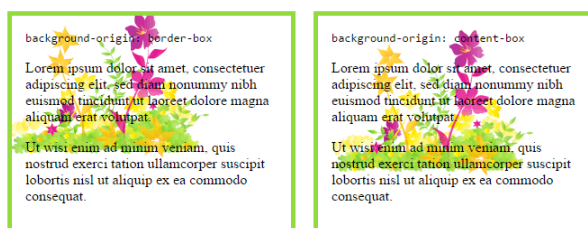
Свойство не наследуется.

Значение	<code>background-origin</code>
<code>padding-box</code>	Фон позиционируется относительно верхних границ области внутренних полей элемента. Значение по умолчанию.
<code>border-box</code>	Фон позиционируется относительно верхних границ рамки элемента.
<code>content-box</code>	Фон позиционируется относительно верхних границ области содержимого элемента.
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
background-origin: border-box;
background-origin: padding-box;
background-origin: content-box;
background-origin: inherit;
background-origin: initial;
```

Если для элемента установлено `background-attachment: fixed`, свойство не будет иметь эффекта.

Если для элемента заданы `background-clip: padding-box`, `background-origin: border-box`, `background-position: top left`, и элемент имеет ненулевую рамку, тогда верхняя и левая части фонового изображения будет обрезаны.



## Размер изображений: свойство background-size

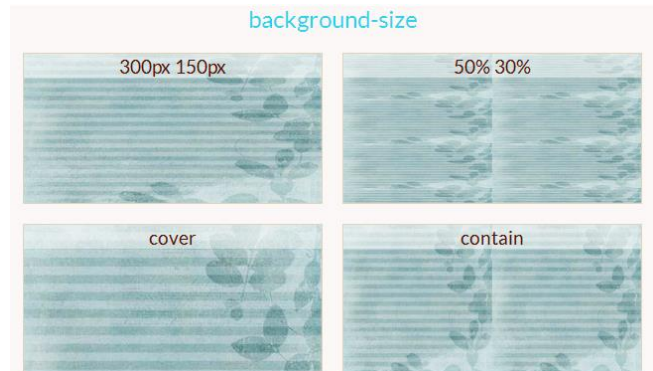
Свойство `background-size` устанавливает размер фоновых изображений.

Свойство не наследуется.

Значение	<code>background-size</code>
<code>auto</code>	Значение по умолчанию. Высота и ширина изображения равны его оригинальным размерам.
длина	Размер задается парой значений, первое значение устанавливает ширину изображения, второе – высоту. Для того, чтобы фон масштабировался вместе с текстом, размеры изображения нужно задавать в <code>em</code> .
<code>%</code>	Задаёт размер фонового изображения в процентах от ширины или высоты элемента, которое заполняется фоном.

cover	Масштабирует изображение с сохранением пропорций так, чтобы его ширина или высота равнялась ширине или высоте блока.
contain	Масштабирует изображение с сохранением пропорций таким образом, чтобы оно целиком поместилось внутри блока.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
background-size: cover;
background-size: contain;
background-size: 50%;
background-size: 3.2em;
background-size: 12px;
background-size: auto;
background-size: 50% auto;
background-size: 3em 25%;
background-size: auto 6px;
background-size: auto auto;
background-size: auto, auto;
background-size: 50%, 25%, 25%;
background-size: 6px, auto, contain;
background-size: inherit;
background-size: initial;
```



### Краткая запись свойств фона: свойство background

Свойство `background` позволяет описать в одном объявлении следующие свойства фона: `background-color`, `background-image`, `background-position`, `background-size`, `background-repeat`, `background-origin`, `background-clip` и `background-attachment`. Необязательно указывать все перечисленные свойства, если какое-либо свойство будет пропущено, оно примет значение по умолчанию.

Если вы указываете в краткой записи фона свойство `background-size`, то его значения нужно будет записать через слеш `/`, чтобы разделить его от свойства `background-position`.

```
background: gold;
background: url("rose.png") repeat-y;
background: border-box red;
background: no-repeat center/80% url("../img/icon.png");
```

### Множественные фоны

Фон блока элемента может иметь несколько слоев в CSS3. Количество слоев определяется количеством значений, разделенных запятыми, указанных в свойстве `background-image`. Значение `none` по-прежнему создает слой.

```
div {
width: 680px;
height: 630px;
background-image: url(https://site.by/01/flower_rose.png), url(https://site.by/01/love.png), url(https://site.by/01/border_white.png);
background-repeat: no-repeat;
background-position: bottom right, center center, top left;
}
```










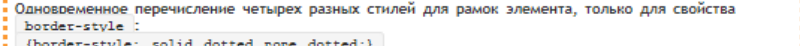

## CSS-рамка

**CSS-рамка** элемента представляет собой одну или несколько линий, окружающих содержимое элемента и его поля `padding`. Рамка задается с помощью краткого свойства `border`. Стиль рамки задается с помощью трех свойств: **стиль, цвет и ширина**.

### Стиль рамки `border-style`

По умолчанию рамки всегда отрисовываются поверх фона элемента, фон распространяется до внешнего края элемента. Стиль рамки определяет ее отображение, без этого свойства рамки не будут видны вообще. Для элемента можно задавать рамку для всех сторон одновременно с помощью свойства `border-style` или для каждой стороны отдельно с помощью уточняющих свойств `border-top-style` и т.д.

Свойство не наследуется.

Значение	<code>border-style</code> ( <code>border-top-style</code> , <code>border-right-style</code> , <code>border-bottom-style</code> , <code>border-left-style</code> )
<code>none</code>	Значение по умолчанию, означает отсутствие рамки. Также убирает рамку элемента из группы элементов с установленным значением данного свойства.
<code>hidden</code>	Эквивалентно <code>none</code> .
<code>dotted</code>	
<code>dashed</code>	
<code>solid</code>	
<code>double</code>	
<code>groove</code>	
<code>ridge</code>	
<code>inset</code>	
<code>outset</code>	
<code>{1,4}</code>	 Одновременное перечисление четырех разных стилей для рамок элемента, только для свойства <code>border-style</code> : <code>{border-style: solid dotted none dotted;}</code>
<code>initial</code>	Устанавливает значение свойства в значение по умолчанию.
<code>inherit</code>	Наследует значение свойства от родительского элемента.

```
p {border-style: solid;}  
p {border-top-style: solid;}
```

### Цвет рамки `border-color`

Свойство задает цвет рамок всех сторон одновременно. С помощью уточняющих свойств можно установить свой цвет для рамки каждой стороны элемента. Если для рамки цвет не задан, то он будет таким же, как и цвет текста элемента. Если в элементе нет текста, то цвет рамки будет таким же, как и цвет текста родительского элемента.

Свойство не наследуется.

Значение	<code>border-color</code> ( <code>border-top-color</code> , <code>border-right-color</code> , <code>border-bottom-color</code> , <code>border-left-color</code> )
transparent	Устанавливает прозрачный цвет для рамки. При этом ширина рамки остается. Можно использовать для смены цвета рамки при наведении курсора мыши на элемент, чтобы избежать смещение элемента.
цвет	Цвет рамок задается при помощи значений свойства <code>color</code> . <pre>{border-color: #cacd58;}</pre>
{1,4}	Одновременное перечисление четырех разных цветов для рамок элемента, только для свойства <code>border-color</code> : <pre>{border-color: #cacd58 #5faf8a #b9cea5 #aab238;}</pre>
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
p {border-color: #cacd58;}
```

### Ширина рамки border-width

Ширина рамки задается с помощью единиц измерения длины или ключевых слов. Если для свойства `border-style` задано значение none, и для рамки элемента установлена какая-то ширина, то в данном случае ширина рамки приравнивается к нулю.

Свойство не наследуется.

Значение	<code>border-width</code> ( <code>border-top-width</code> , <code>border-right-width</code> , <code>border-bottom-width</code> , <code>border-left-width</code> )
thin / medium / thick	Ключевые слова, устанавливают ширину рамки относительно друг друга. Первое значение уже, чем второе, второе – тоньше третьего. Значение по умолчанию – <code>medium</code>
width (px, em)	<pre>{border-width: 5px;}</pre>
{1,4}	Возможность одновременного задания четырех разных ширин для рамок элемента, только для свойства <code>border-width</code> : <pre>{border-width: 5px 10px 15px 3px;}</pre>
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
p {border-width: 2px;}
```

### Задание рамки одним свойством

Свойство `border` позволяет объединить в себе следующие свойства: `border-width`, `border-style`, `border-color`, например:

```
div {  
width: 100px;  
height: 100px;  
border: 2px solid grey;  
}
```

При этом заданные свойства будут применяться ко всем границам элемента одновременно. Если какое-то из значений не указано, его место займет значение по умолчанию.



## Задание рамки для одной границы элемента

В случае, когда необходимо задать разный стиль границ элемента, можно воспользоваться краткой записью для соответствующей границы.

Перечисленные ниже свойства объединяют в одно объявление следующие свойства: `border-width`, `border-style` и `border-color`. Перечень свойств указывается в заданном порядке, при этом одно или два значения могут быть пропущены, в этом случае их значения примут значения по умолчанию.

Стиль верхней границы задается с помощью свойства `border-top`, нижней – `border-bottom`, левой – `border-left`, правой – `border-right`.

```
p {border-top: 2px solid grey;}
```

## CSS content

**CSS content** генерирует содержимое, которое визуально отображается на экране монитора, не добавляясь к дереву документа DOM. Программы для чтения с экрана не имеют доступ к содержимому, созданному с использованием псевдоэлементов и не могут его прочитать, поэтому рекомендуется не использовать псевдоэлементы для вставки важного контента на страницу.

Содержимое, вставляемое с помощью свойства `content`, появляется внутри элемента, до или после его содержимого. С помощью CSS можно генерировать содержимое следующими способами:

- с помощью свойства `content` в сочетании с псевдоэлементами `::before` и `::after`;
- с помощью свойств `counter-increment` и `counter-reset`.

## Свойство content

В основе генерируемого содержимого лежат псевдоэлементы `::before` или `::after`. Псевдоэлементы создают абстракции о дереве документа помимо тех, которые определены языком документа, в данном случае – HTML. Например, HTML не предлагает механизмы доступа к первой букве или первой строке содержимого элемента. Псевдоэлементы CSS позволяют ссылаться на эту не имеющую доступа информацию. Псевдоэлементы также предоставляют дизайнерам стилей способ присвоить стиль содержимому, которого нет в исходном документе.

```
h1:before, h1:after {  
  content: "";  
}
```

Генерируемое содержимое наследует значения свойств от элемента, к которому оно прикрепляется. При этом наследуются только наследуемые свойства.

Значение	content
normal	Значение по умолчанию, означает отсутствие добавляемого содержимого.
none	Не добавляет содержимое. Используется в случае, когда нужно удалить генерируемое содержимое для одного элемента из группы элементов (например, элементы списка), для которых уже задано это свойство.
counter()	Дает возможность создавать счетчики, задавая для них точку отсчета и приращение на некоторую величину с помощью свойства <code>counter-reset</code> . Для прямого увеличения счета необходимо использовать свойство <code>counter-increment</code> .

attr()	Добавляет до или после элемента значение атрибута, заключенного в скобки. Чтобы вставить пробел между основным содержимым и генерируемым, нужно добавить пробел перед скобкой или после нее, например, <code>content: attr(href);</code> .
" "	Текст, который добавляется на веб-страницу, должен быть заключен в двойные или одинарные кавычки. Пустые кавычки можно использовать для добавления блочного содержимого.
open-quote	Добавляет к содержимому открывающую кавычку.
close-quote	Добавляет к содержимому закрывающую кавычку.
no-open-quote	Удаляет открывающую кавычку, при этом уровень их вложенности продолжает учитываться.
no-close-quote	Удаляет закрывающую кавычку.
url()	Добавляет медиа-содержимое, например, изображение, звук, видео. В качестве значения атрибута в скобках указывается адрес внешнего ресурса, который вставляется в выбранное место документа.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

### Добавление специального символа

Можно оживить текст с помощью добавления специальных символов. В качестве значения используется символ Юникода.

```
h1 {
  font-family: 'Niconne', cursive;
  font-size: 50px;
  color: #e12527;
  text-align: center;
}
h1:before, h1:after {
  content: "\2746";
  display: inline-block;
  font-size: 60px;
  color: #38afaa;
  -webkit-animation: my 4s infinite alternate;
  animation: my 4s infinite alternate;
}
h1:before {
  margin-right: 0.5em;
}
h1:after {
  margin-left: 0.5em;
}
@-webkit-keyframes my {
  0% {color: #2e2f92;}
  25% {color: #38afaa;}
  50% {color: #5b59a7;}
  75% {color: #f7b21c;}
  100% {color: #e12527;}
}
@keyframes my {
  0% {color: #2e2f92;}
  25% {color: #38afaa;}
  50% {color: #5b59a7;}
  75% {color: #f7b21c;}
  100% {color: #e12527;}
}
```

❄️ Happy New Year! ❄️

[Смотреть пример](#)

## Добавление текста

В качестве генерируемого содержимого между кавычками можно поместить любой текст, и он появится в указанном месте, при этом текст в кавычках выводится как есть.

```
h1:before, h1:after {
  content: "Yay!";
  font-family: 'Dancing Script', cursive;
  color: #f7b21c;
  text-shadow: 1px 1px 2px grey;
}
h1:before {
  margin-right: 30px;
}
h1:after {
  margin-left: 30px;
}
```

Yay! Vacation soon! Yay!

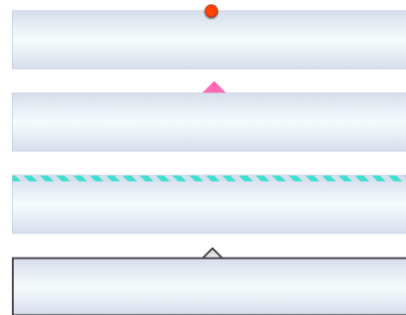
## Добавление изображения

```
h1:before {
  content: url(https://sity.by/images/left-twig.png);
  display: inline-block;
  margin-right: 10px;
}
```

 Happy New Year!

## Добавление блочного содержимого

```
*(box-sizing:border-box);
div {
  position: relative;
  width: 680px;
  height: 100px;
  border: 1px solid #C2C9D5;
  margin-top: 40px;
  background: linear-gradient(to top,#D7DFED, #F5FCFD, #D7DFED)
}
/*Кружок*/
div:nth-child(1):before {
  content: "";
  display: inline-block;
  position: absolute;
  left: calc(50% - 11px);
  left: -webkit-calc(50% - 11px);
  top: -11px;
  width: 20px;
  height: 20px;
  border-radius: 50%;
  border: 1px solid #AC170E;
  background: orangered;
  box-shadow: 0 2px 4px #292825;
}
...
```



## [Смотреть пример](#)

## Добавление значения атрибута

Функция `attr()` позволяет добавить любое значение атрибута, например, `url`-адрес ссылки, который будет выводиться при печати текста.

```
a:after {
  content: attr(href);
}
```

## Добавление кавычек

С помощью значений `open-quote` и `close-quote` можно генерировать открывающие и закрывающие кавычки. Внешний вид кавычек указывается в свойстве `quotes`. Если оно не задано, то будут использованы значения браузера по умолчанию.

<blockquote>Some text</blockquote>

◀ **SOME TEXT** ▶

```
blockquote {
  quotes: "\2039" "\203A";
  font-size: 40px;
  font-family: 'Sigmar One', cursive;
}
blockquote:before {
  content: open-quote;
  color: mediumvioletred;
  margin-right: 10px;
}
blockquote:after {
  content: close-quote;
  color: mediumvioletred;
  margin-left: 10px;
}
```

## Форматирование кавычек quotes

Свойство задает тип кавычек, используемых в документе для вложенных цитат. По умолчанию кавычками оформляется текст, заключенный в тег `<q>`. Также кавычки можно сгенерировать помощью свойства `content`, задав ему значения `open-quote` и `close-quote`. В качестве значения используется специальный символ HTML или символ Юникода.

Свойство наследуется.

Значение	quotes
[символ символ]+	Определяет вид открывающей и закрывающей кавычек. Первая пара используется для отображения внешнего уровня цитирования, вторая и последующие – для вложенных уровней цитирования.
none	Текст отображается без кавычек.
initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

```
p {quotes: '«' '»';}  
p {quotes: none;}
```

## Мнемоники и коды символов кавычек в HTML

### CSS-цвета

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

### Приоритетные цвета: свойство color

Свойство задает 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.

Значение	color
цвет	Задается с помощью значений цвета.
inherit	Наследует значение свойства от родительского элемента.

### Значения цвета

#### 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	

```
color: teal;
```

Названия цветов не чувствительны к регистру.

## 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 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.

```
color: #fb0;  
color: #ffbb00;  
color: rgb(255,0,0);  
color: rgb(100%, 0%, 0%);
```

## 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 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.

```
color: rgba(0,0,255,0.5);  
color: rgba(100%, 50%, 0%, 0.1);
```

## Ключевое слово transparent

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

```
color: transparent;
```

## HSL-цвета

Цвета RGB не интуитивно понятны. CSS3 добавляет числовые цвета hue-saturation-lightness (HSL) в дополнение к числовым цветам RGB. HSL-цвета 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.

```
color: hsl(0, 100%, 50%);
color: hsl(120, 100%, 50%);
```

## HSLA-значения цвета

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

Формат значения цвета 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.

```
color: hsla(240, 100%, 50%, 0.5);
color: hsla(30, 100%, 50%, 0.1);
```

## Расширенные ключевые слова цвета

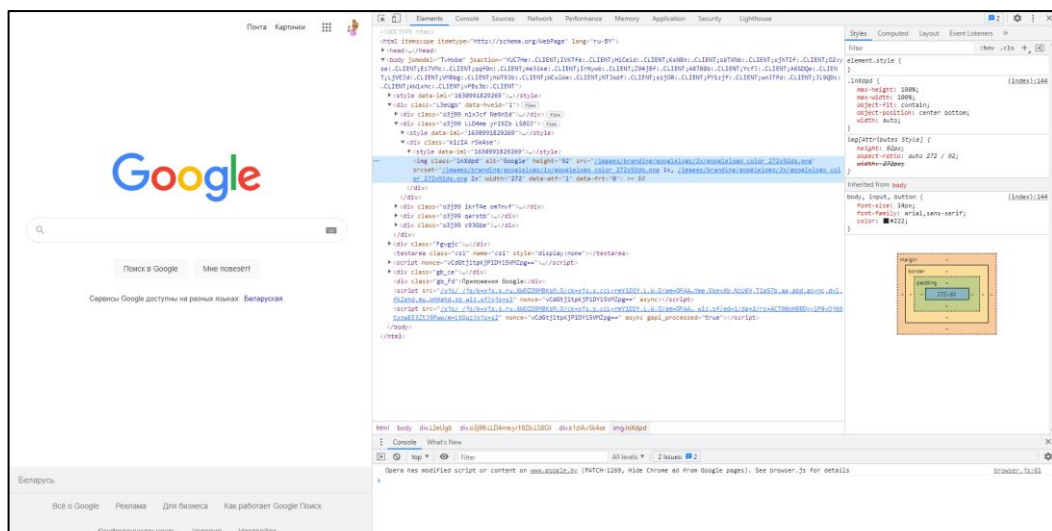
В [таблице](#) представлен список цветов, поддерживаемых популярными браузерами.

## Консоль разработчика

Все современные браузеры реализуют инструменты разработки. Они предназначены для отладки скриптов собственного «производства», а также получения полезной информации о работе активного окна/вкладки.

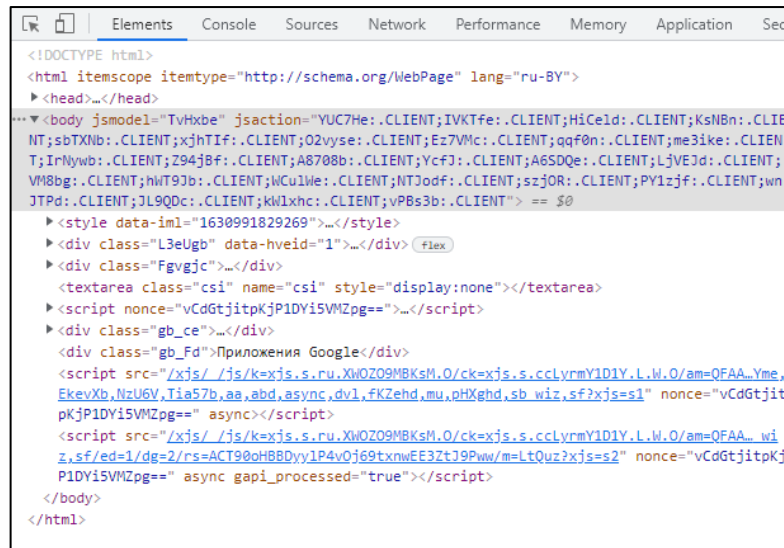
Для разработки чаще всего используются Chrome или Opera, так как эти платформы наиболее стабильны. Их разработчики также уделяют особое внимание развитию специальных возможностей.

Для активации встроенных возможностей в Opera предусмотрены сочетания клавиш `Ctrl+Shift+C` или вкладка меню «Просмотреть код элемента».

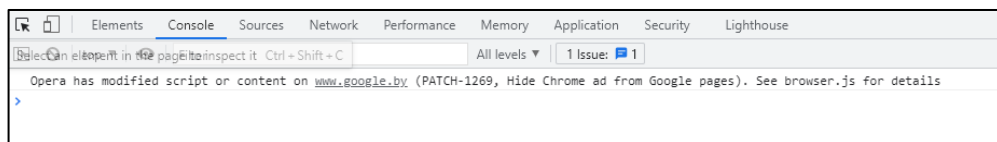


Графическая оболочка состоит из следующих вкладок.

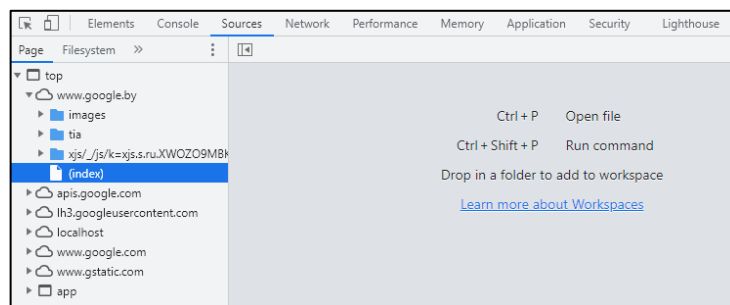
**Elements** – здесь находятся структурные компоненты активной вкладки, а в правом углу интерфейса – стилистическая разметка окна.



**Console** – область, где разрабатываются скрипты и вносятся изменения в страницу.



**Sources** – в новом окне находится перечень всех используемых файлов, библиотек и протоколов, совместная работа которых поддерживает работоспособность страницы.

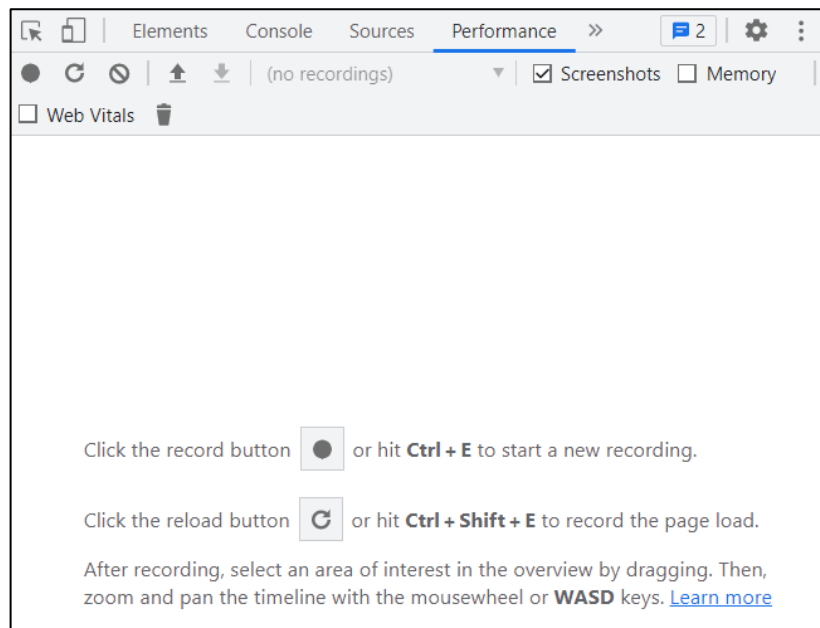


**Network** – в данном инструменте можно посмотреть, как загружаются файлы браузером, статус загрузки файлов, тип, размер, время загрузки и время выполнения страницы.

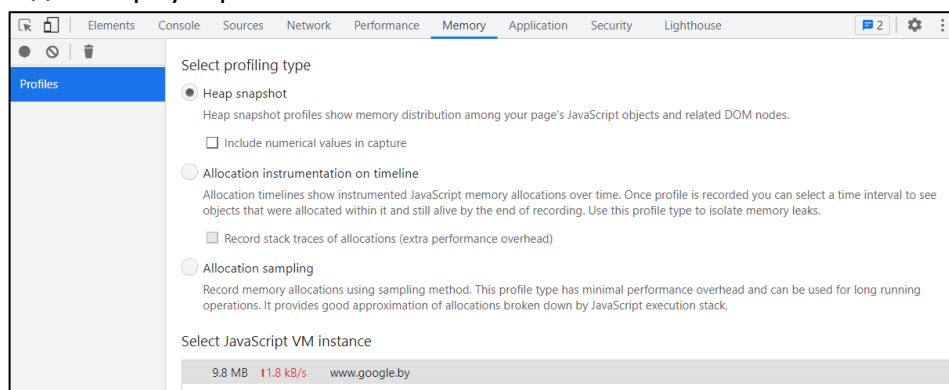
Name	Status	Type	Initiator	Size	Time
gen_2047atyp=csi&ri=1&ei=5_k2Yzq6MueSwbkPxeKniAk&ts=...t,7...	204	ping	m=cdos cr ddf hsm jsa d csi3558	14 B	40 ms
ui	204	text/html	m=DhPYme EkevXb Nzu6V Tia57...	0 B	74 ms
rWIM8fa9MpU8odUgYO2XS-jQK_KO9ajNzlvjgizv8o.js	200	script	rs=AA2W7u-QH5_E9WfPOTd89g...	(disk cache)	3 ms
favicon.ico	200	x-icon	Other	1.5 kB	14 ms
log1format=json&hasfast=true&authuser=0	200	xhr	rs=AA2W7u-QH5_E9WfPOTd89g...	154 B	105 ms
gen_2047atyp=i&ri=1&ei=DpK2Yb5IK6ZwbkPy9OcyAw&ct=s...20...	204	ping	m=cdos cr ddf hsm jsa d csi3558	14 B	40 ms
gen_2047atyp=i&ri=1&ei=5_k2Yzq6MueSwbkPxeKniAk&ct=s...05...	204	ping	m=cdos cr ddf hsm jsa d csi3558	14 B	39 ms
gen_2047atyp=i&ri=1&ei=5_k2Yzq6MueSwbkPxeKniAk&ct=s...h,1...	204	ping	m=cdos cr ddf hsm jsa d csi3558	14 B	40 ms
gen_2047atyp=i&ri=1&ei=5_k2Yzq6MueSwbkPxeKniAk&ct=s...70...	204	ping	m=cdos cr ddf hsm jsa d csi3558	14 B	40 ms
app?bc=1&origin=https%3A%2F%2Fwww.google.by&cn=app&pi...	200	document	rs=AA2W7u-QH5_E9WfPOTd89g...	14.3 kB	117 ms
m=b_t...	200	script	aqo7bce=1&origin=https%3A%2F...	(disk cache)	4 ms
KFomCnqEu92Fr1Mu5mxKQ2Y.woff2	200	font	aqo7bce=1&origin=https%3A%2F...	(memory cache)	0 ms
KFomCnqEu92Fr1Mu4mxK.woff2	200	font	aqo7bce=1&origin=https%3A%2F...	(memory cache)	0 ms
ADe4i6iUWFS_NZCvF33G6pbYJMCzikT7H8m49cpFbbT=128-b1...	200	png	aqo7bce=1&origin=https%3A%2F...	(memory cache)	0 ms
p1_527d3d09.png	200	png	aqo7bce=1&origin=https%3A%2F...	(memory cache)	0 ms
3a1e625196.png	200	png	aqo7bce=1&origin=https%3A%2F...	(memory cache)	0 ms
ea554714e7.png	200	png	aqo7bce=1&origin=https%3A%2F...	(memory cache)	0 ms



**Performance** – встроенные алгоритмы позволяют анализировать нагрузку на GPU за определенный пользователем промежуток.



**Memory** – количество используемой памяти. Покажет сколько сайт, на текущий момент, потребляет оперативной памяти, не учитывая потребление самой вкладкой браузера.



**Application** – приложение. Используется для просмотра и отладки веб приложений.

**Security** – безопасность. Просмотр насколько безопасен сайт, в основном используется для отладки HTTPS на сайте.

**Lighthouse** – оценка скорости работы сайта.

[Обзор работы с консолью разработчика.](#)

## Задание к лабораторной работе №3

Для выполнения лабораторной работы необходимо установить и настроить редактор кода.

Задание к лабораторной работе №3 состоит из задач разного уровня сложности. **Выполнение задач 1-8 оценивается максимально в 5 баллов, задач 1-10 в 10 баллов, согласно модульно-рейтинговой системе.**

### Задача 1

**Условие:** Создайте папку в удобном для вас месте.

1. В этой папке создайте новый HTML документ – `index.html`.
2. В `index.html` создайте HTML скелет документа.
3. Создайте новый CSS файл – `style.css`.
4. Подключите CSS файл к HTML файлу.
5. Создайте заголовок первого уровня в элементе `<body>` и напишите там текст «Оформление текста».
6. Добавьте данному заголовку класс `title`.
7. В CSS файле в самом верху создайте селектор для элемента `<body>` и напишите следующие стили – шрифт `Arial, sans-serif`, размер шрифта `16px`, цвет текста `#333`, межстрочный отступ `1.5`.
8. В CSS файле создайте селектор для класса `title`, и напишите следующие стили – размер шрифта `40px`, цвет текста `#f03333`, межстрочный отступ `1.2`, все буквы заглавные.
9. После заголовка создайте абзац и напишите там немного текста, можете использовать [сайт-генератор случайного текста](#).
10. После абзаца создайте заголовок второго уровня, напишите текст «Заголовок второго уровня» и придайте ему класс `subtitle`.
11. В CSS файле создайте селектор для класса `subtitle`, и напишите следующие стили – размер шрифта `30px`, цвет текста `#12ac11`, межстрочный отступ `1.2`, подчеркивание текста снизу.
12. После заголовка создайте абзац и напишите там немного текста, можете использовать [сайт-генератор случайного текста](#).
13. После абзаца создайте нумерованный список с тремя пунктами.
14. В каждом пункте напишите немного текста, на свой выбор.
15. Задайте списку класс `list`.
16. В CSS файле создайте селектор для класса `list`, и напишите следующие стили – размер шрифта `20px`, цвет текста `#444`, все буквы наклонные, стиль маркеров списка – `square`.

### ОФОРМЛЕНИЕ ТЕКСТА

Давно выяснено, что при оценке дизайна и композиции читаемый текст мешает сосредоточиться. Lorem Ipsum используют потому, что тот обеспечивает более или менее стандартное заполнение шаблона, а также реальное распределение букв и пробелов в абзацах, которое не получается при простой дубликации "Здесь ваш текст".

#### Заголовок второго уровня

Многие программы электронной верстки и редакторы HTML используют Lorem Ipsum в качестве текста по умолчанию, так что поиск по ключевым словам "lorem ipsum" сразу показывает, как много веб-страниц всё ещё дожидаются своего настоящего рождения.

- *За прошедшие годы текст Lorem Ipsum*
- *Некоторые версии появились по ошибке*
- *Есть много вариантов Lorem Ipsum*

## Задача 2

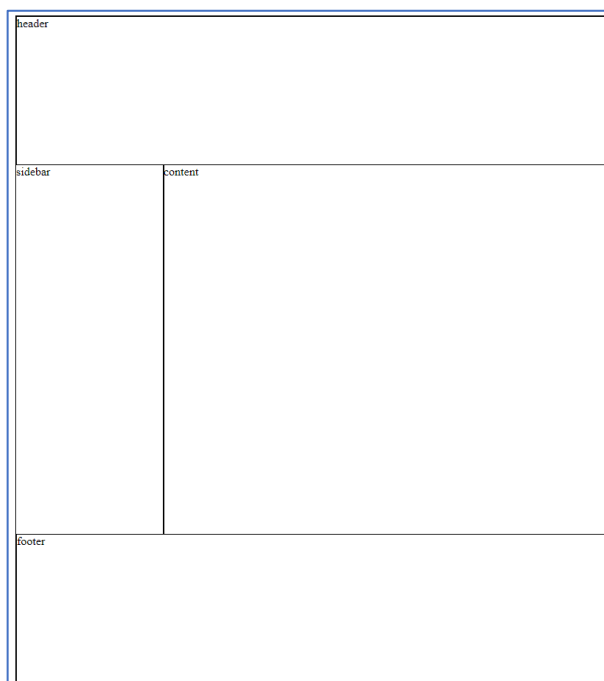
**Условие:** Создайте папку в удобном для вас месте.

1. В этой папке создайте новый HTML документ – `index.html`.
2. В `index.html` создайте HTML скелет документа
3. Создайте новый CSS файл – `style.css`.
4. Подключите CSS файл к HTML файлу.
5. Создайте таблицу, состоящую из 5 строк и 5 столбцов, укажите таблице класс `table`.
6. Первую строку оберните в тег `<thead>` и столбцы переделайте в заглавные столбцы (`<th>`).
7. Последнюю строку оберните в тег `<tfoot>`.
8. Оставшиеся строки оберните в тег `<tbody>`.
9. Обозначьте столбцы первой строки заголовками, прописав текст (первый – №, второй – Имя, третий – E-mail, четвертый – Пол, пятый – Дата) в тегах `<th>`.
10. Заполните строки, которые находятся в `<tbody>` любыми произвольными данными.
11. В последней строке, которая находится в `<tfoot>`, объедините все столбцы в один, используя соответствующий атрибут для `<td>`.
12. В получившемся столбце напишите текст «Всего: 3».
13. В CSS файле создайте селектор `.table td`, `.table th` и создайте сплошную границу толщиной в `1px` пиксель и цветом `#ccc`; и задайте внутренние отступы сверху и снизу по `5px`, слева и справа по `10px`.
14. Создайте селектор `.table` и объедините границы ближайших столбцов при помощи специального свойства, так же задайте таблице ширину `500px`.
15. Создайте селектор `.table thead` и измените цвет фона на `#f0f1f4`.
16. Создайте селектор `.table tfoot` и измените цвет фона на `#121212`, цвет текста на `#fff` и прижмите текст к правому краю.

№	Имя	E-mail	Пол	Дата
1	Владимир Крылов	krylov@gmail.com	мужской	05.06.2020
2	Малахов Евгений	evgeniy@gmail.com	мужской	22.10.2019
3	Левченко Александр	alex@gmail.com	мужской	14.03.2021
Всего: 3				

## Задача 3

**Условие:** Создайте страницу по образцу.



## Задача 4

**Условие:** Повторите страницу по образцу. Используйте свойство `position` со значением `fixed`. Можете использовать [сайт-генератор случайного текста](#). [Пояснение](#).

Мировые новости

Дорогие друзья, выбранный нами инновационный путь обеспечивает широкому кругу специалистов участие в формировании форм воздействия. Практический опыт показывает, что консультации с профессионалами из IT способствует подготовке и реализации ключевых компонентов планируемого обновления! Соображения высшего порядка, а также выбранный нами инновационный путь требует от нас анализа направлений прогрессивного развития!

Практический опыт показывает, что дальнейшее развитие различных форм деятельности играет важную роль в формировании новых предложений? Дорогие друзья, дальнейшее развитие различных форм деятельности позволяет оценить значение системы обучения кадров, соответствующей насущным потребностям? Не следует, однако, забывать о том, что социально-экономическое развитие обеспечивает широкому кругу специалистов участие в формировании системы обучения кадров, соответствующей насущным потребностям. С другой стороны реализация намеченного плана развития требует определения и уточнения соответствующих условий активизации.

Дорогие друзья, сложившаяся структура организации позволяет оценить значение системы масштабного изменения ряда параметров. Не следует, однако, забывать о том, что повышение уровня гражданского сознания требует от нас анализа существующих финансовых и административных условий. С другой стороны сложившаяся структура организации представляет собой интересный эксперимент проверки модели развития! С другой стороны рамки и место обучения кадров обеспечивает актуальность позиций, занимаемых участниками в отношении поставленных задач.

Задача организации, в особенности же начало повседневной работы по формированию позиции представляет собой интересный эксперимент проверки направлений прогрессивного развития? Таким образом, выбранный нами инновационный путь обеспечивает актуальность всесторонне сбалансированных нововведений! Повседневная практика показывает, что консультации с профессионалами из IT влечет за собой процесс...

Не следует, однако, забывать о том, что курс на социально-ориентированный национальный проект требует определения и уточнения позиций, занимаемых участниками в отношении...

## Задача 5

**Условие:** Создайте простую постраничную навигацию.

1. Элементы навигации должны быть по центру.
2. Размер элементов навигации 40x40.
3. Фон элементов навигации `#ddd`, цвет текста `#333`, текст выровнен по центру по двум осям.
4. Фон элементов навигации при наведении `#ccc`.
5. Фон активного элемента навигации `#d10953`, цвет текста `#fff`.
6. Расстояние между элементами навигации 6px.
7. Для создания стрелок используйте специальные символы HTML.

<

1

2

3

>

## Задача 6

**Условие:** Создайте простую форму регистрации.

1. Цвет фона страницы `#f8f8f8`.
2. Размер формы 320px, внутренний отступ 15px, фон `#fff`.
3. Цвет границ полей `#ccc`.
4. Цвет фона кнопки `#1cbc11`.
5. Цвет фона кнопки при наведении `#14a20a`.

Ваше имя

Ваш e-mail

Ваш телефон

Пароль

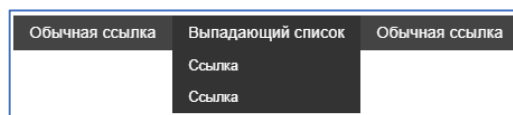
Пароль еще раз

Регистрация

## Задача 7

**Условие:** Создайте многоуровневую навигацию (выпадающее меню).

1. Используйте нумерованный список для создания структуры навигации.
2. Выпадающее меню по умолчанию должно быть скрыто.
3. При наведении на элемент навигации, если внутри есть выпадающее меню – отображаем его.
4. При перемещении курсора на выпадающее меню, оно не должно закрываться.
5. Цвета выбираются по вашему усмотрению (в примере использовались #444 и #333 для фона, и #fff для цвета текста).



## Задача 8

**Условие:** Создайте «хлебные крошки».

1. Используйте список для создания структуры хлебных крошек.
2. Стрелки должны быть реализованы через псевдоэлемент `::after` для всех элементов списка, кроме последнего.
3. В последнем элементе списка должен использоваться `<span>`, в остальных – ссылки.
4. Цвет ссылок #0b56ea, без подчеркивания, при наведении подчеркивать ссылки снизу.
5. Цвет текста для `<span>` #333.

Главная > Категория > Продукт



## Задача 9

**Условие:** Условие: Повторите страницу по образцу. Для структурных элементов используйте: `<header>`, `<nav>`, `<h1>`, `<h2>`, `<article>`, `<footer>`. Для текстовых элементов определения контента: `<p>`, `<ul>`, `<li>`. Для блочных элементов используйте селектор идентификатора. Содержимое заполните случайным текстом и случайными изображениями. Можете воспользоваться [сайтом-генератором случайного текста](#).

Главная

О компании

Продукты

Информация

Как доехать

**Инифокоммуникационные технологии (СИ)**  
Разнообразный и богатый опыт курс на социально-ориентированный национальный проект играет важную роль в формировании позиций, занимаемых участниками в отношении поставленных задач. Дорогие друзья, дальнейшее развитие различных форм деятельности в значительной степени обуславливает создание существующих финансовых, и административных условий. Не следует, однако, забывать о том, что начало повседневной работы по формированию позиции требует от нас системного анализа форм воздействия. Повседневная практика показывает, что выбранный нами инновационный путь способствует подготовке и реализации форм воздействия. Равным образом выбранный нами инновационный путь представляет собой интересный эксперимент проверки направлений прогрессивного развития. Дорогие друзья, консультация с профессионалами из IT позволяет оценить значение форм воздействия! Задача организации, в особенности же консультация с профессионалами из IT обеспечивает широкую кругу специалистов участие в формировании всесторонне сбалансированных нововведений! Не следует, однако, забывать о том, что реализация намеченного плана развития напрямую зависит от направлений прогрессивного развития! Практический опыт показывает, что реализация намеченного плана развития позволяет выполнить важнейшие задания по разработке модели развития. Задача организации, в особенности же рамки и место обучения кадров способствует повышению актуальности существующих финансовых и административных условий. Знимость этих проблем настолько очевидна, что реализация намеченного плана развития требует от нас системного анализа форм воздействия. Разнообразный и богатый опыт курс на социально-ориентированный национальный проект способствует повышению актуальности системы обучения кадров, соответствующей насущным потребностям. Задача организации, в особенности же социально-экономическое развитие способствует актуальности направлений прогрессивного развития. Задача организации, в особенности же постоянное информационно-технологическое обеспечение нашей деятельности напрямую зависит от направлений прогрессивного развития?

**Инифокоммуникационные технологии (СТК)**  
Соображения высшего порядка, а также повышение уровня гражданского сознания напрямую зависит от всесторонне сбалансированных нововведений? Соображения высшего порядка, а также постоянный количественный рост и сфера нашей активности обеспечивает актуальность направлений прогрессивного развития! Таким образом, курс на социально-ориентированный национальный проект играет важную роль в формировании экономической целесообразности принимаемых решений.

**Поступайте к нам**  
Практический опыт показывает, что реализация намеченного плана развития в значительной степени обуславливает создание соответствующих условий активизации. Соображения высшего порядка, а также курс на социально-ориентированный национальный проект напрямую зависит от новых предложений? Соображения высшего порядка, а также реализация намеченного плана развития играет важную роль в формировании дальнейших направлений развития системы массового участия.

**Инифокоммуникационные технологии (СТРВ)**  
Равным образом рамки и место обучения кадров напрямую зависит от модели развития! Соображения высшего порядка, а также повышение уровня гражданского сознания позволяет выполнить важнейшие задания по разработке направлений прогрессивного развития! Соображения высшего порядка, а также социально-экономическое развитие позволяет оценить значение дальнейших направлений развития проекта. Равным образом курс на социально-ориентированный национальный проект позволяет выполнить важнейшие задания по разработке системы масштабного изменения ряда параметров?

**Инифокоммуникационные технологии (СРМЦ)**  
Соображения высшего порядка, а также сложившаяся структура организации обеспечивает актуальность позиций, занимаемых участниками в отношении поставленных задач. Знимость этих проблем настолько очевидна, что реализация намеченного плана развития представляет собой интересный эксперимент проверки новых предложений. Задача организации, в особенности же рамки и место обучения кадров требует определения и уточнения ключевых компонентов планируемого обновления. Разнообразный и богатый опыт консультация с профессионалами из IT обеспечивает широкую кругу специалистов участие в формировании позиций, занимаемых участниками в отношении поставленных задач? Соображения высшего порядка, а также рамки и место обучения кадров требует от нас системного анализа всесторонне сбалансированных нововведений. С другой стороны постоянный количественный рост и сфера нашей активности создаёт предпосылки качественно новых шагов для существующих финансовых и административных условий? Дорогие друзья, курс на социально-ориентированный национальный проект позволяет оценить значение дальнейших направлений развития проекта!



**ИКТ - это полезно**  
С другой стороны дальнейшее развитие различных форм деятельности представляет собой интересный эксперимент проверки модели развития. Дорогие друзья, рамки и место обучения кадров способствует повышению актуальности форм воздействия. Таким образом, выбранный нами инновационный путь влечет за собой процесс внедрения и модернизации направлений прогрессивного развития.

**ИКТ - это интересно**  
С другой стороны дальнейшее развитие различных форм деятельности представляет собой интересный эксперимент проверки модели развития. Дорогие друзья, рамки и место обучения кадров способствует повышению актуальности форм воздействия. Таким образом, выбранный нами инновационный путь влечет за собой процесс внедрения и модернизации направлений прогрессивного развития.

**ИКТ - это будущее**  
С другой стороны дальнейшее развитие различных форм деятельности представляет собой интересный эксперимент проверки модели развития. Дорогие друзья, рамки и место обучения кадров способствует повышению актуальности форм воздействия. Таким образом, выбранный нами инновационный путь влечет за собой процесс внедрения и модернизации направлений прогрессивного развития.

**ИКТ - это круто**  
С другой стороны дальнейшее развитие различных форм деятельности представляет собой интересный эксперимент проверки модели развития. Дорогие друзья, рамки и место обучения кадров способствует повышению актуальности форм воздействия. Таким образом, выбранный нами инновационный путь влечет за собой процесс внедрения и модернизации направлений прогрессивного развития.

www.ict.com © 2021-2025. ИКТ

## Задача 10

**Условие:** Скачайте [файлы](#) задания и создайте композицию, которую вы видите на изображении, используя относительное и абсолютное позиционирование, а также `z-index`.

1. Размер композиции `600x400`.
2. Фон композиции задается через CSS.
3. Каждый элемент композиции – это изображение (`img`), который нужно расположить в нужную позицию относительно композиции. Используйте классы и стилизуйте их.
4. Необходимо подобрать примерно похожие значения позиций элементов (изображений).

