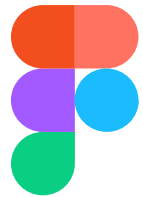


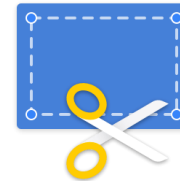
Подготовка рабочего места



Visual Studio Code



Figma



CODEPEN



Ссылки на ресурсы

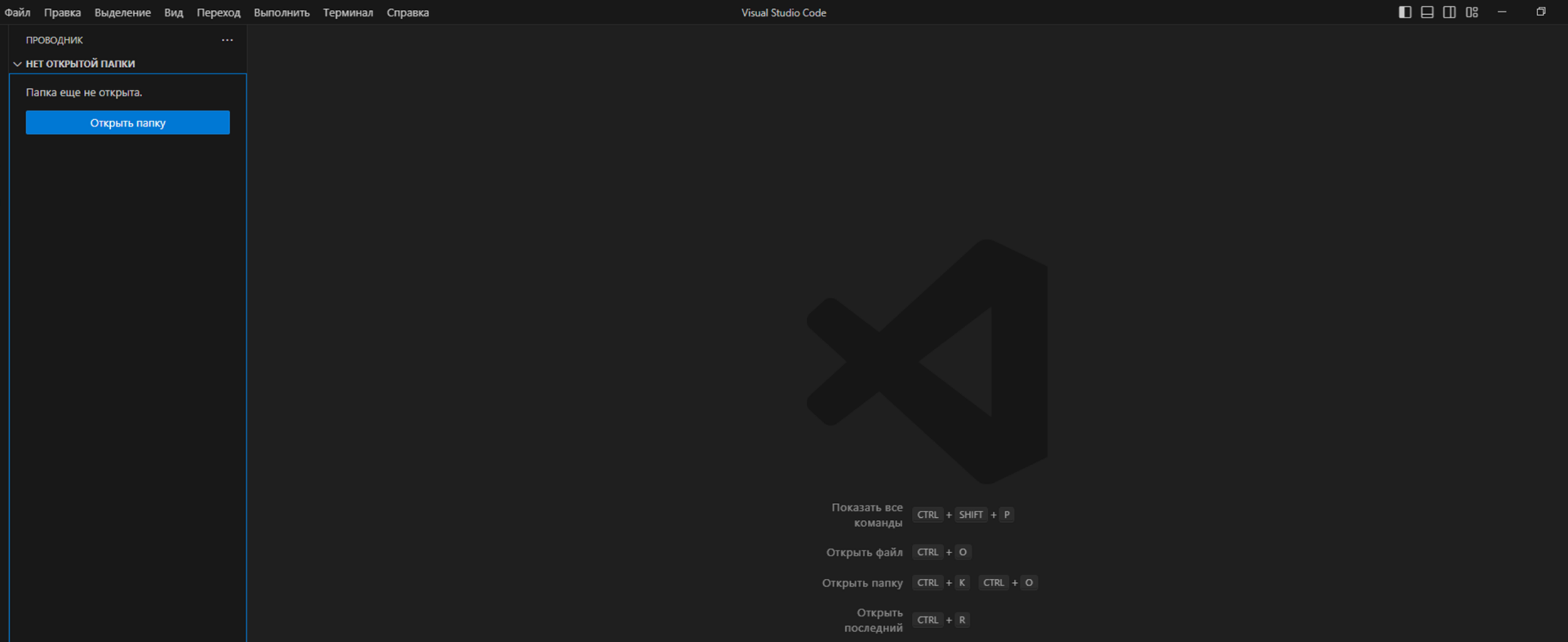
VS Code <https://code.visualstudio.com/>

Figma <https://www.figma.com>

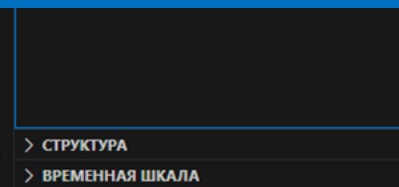
Codepen <https://codepen.io/>

Github <https://github.com/>


Справочник <https://webref.ru/>






VS Code




Полезные плагины VS Code




Russian Language Pack for Visual Studio Code v1.64.7

 Microsoft |
  1 745 525 |
  (16)




Language pack extension for Russian

[Удалить](#)



Это расширение включено на глобальном уровне.




Live Server v5.7.4

 Ritwick Dey |
  18 915 845 |
  (357)




Launch a development local Server with live reload feature for static & dynamic pages

[Отключить](#)
[Удалить](#)



Это расширение включено на глобальном уровне.




Path Intellisense v2.8.0

 Christian Kohler |
  6 423 154 |
  (102)




Visual Studio Code plugin that autocompletes filenames

[Отключить](#)
[Удалить](#)



Это расширение включено на глобальном уровне.



Auto Rename Tag v0.1.10

 Jun Han |
  8 083 142 |
  (150)

Auto rename paired HTML/XML tag

[Отключить](#)
[Удалить](#)


Это расширение включено на глобальном уровне.









Image preview v0.30.0

 Kiss Tamás |
  687 897 |
  (37)




Shows image preview in the gutter and on hover

[Отключить](#)
[Удалить](#)



Это расширение включено на глобальном уровне.



eCSStractor for VSCode v0.0.3

 diz |
  130 117 |
  (9)

VSCode plugin for extracting class names from HTML and generate CSS st...

[Отключить](#)
[Удалить](#)


Это расширение включено на глобальном уровне.

команды

CTRL + SHIFT + P

CTRL +

CTRL +

CTRL +

Введение в HTML

HTML

- Язык разметки (нет алгоритмических составляющих)
- Создание каркаса страницы
- Основная структурная единица - тег

Тег

HTML-теги — основа языка HTML. Теги используются для разграничения начала и конца элементов в разметке.

У тега могут быть атрибуты (настройки)

Теги:

- Парные
- Одиночные



Комбинация начального тега, конечного тега и содержимого называется **HTML-элементом**

Теги



```
1 <body>
2     <div>
3         <p>
4             Привет!
5         </p>
6     </div>
7 </body>
8
```


Теги языка HTML можно условно разделить на несколько типов:

Теги для **форматирования** текста и указания ссылок: h1 - h6, b, em, ul, ol, a, и т.д.

Теги **структуры** документа: div, span, nav, header, section и т.д.

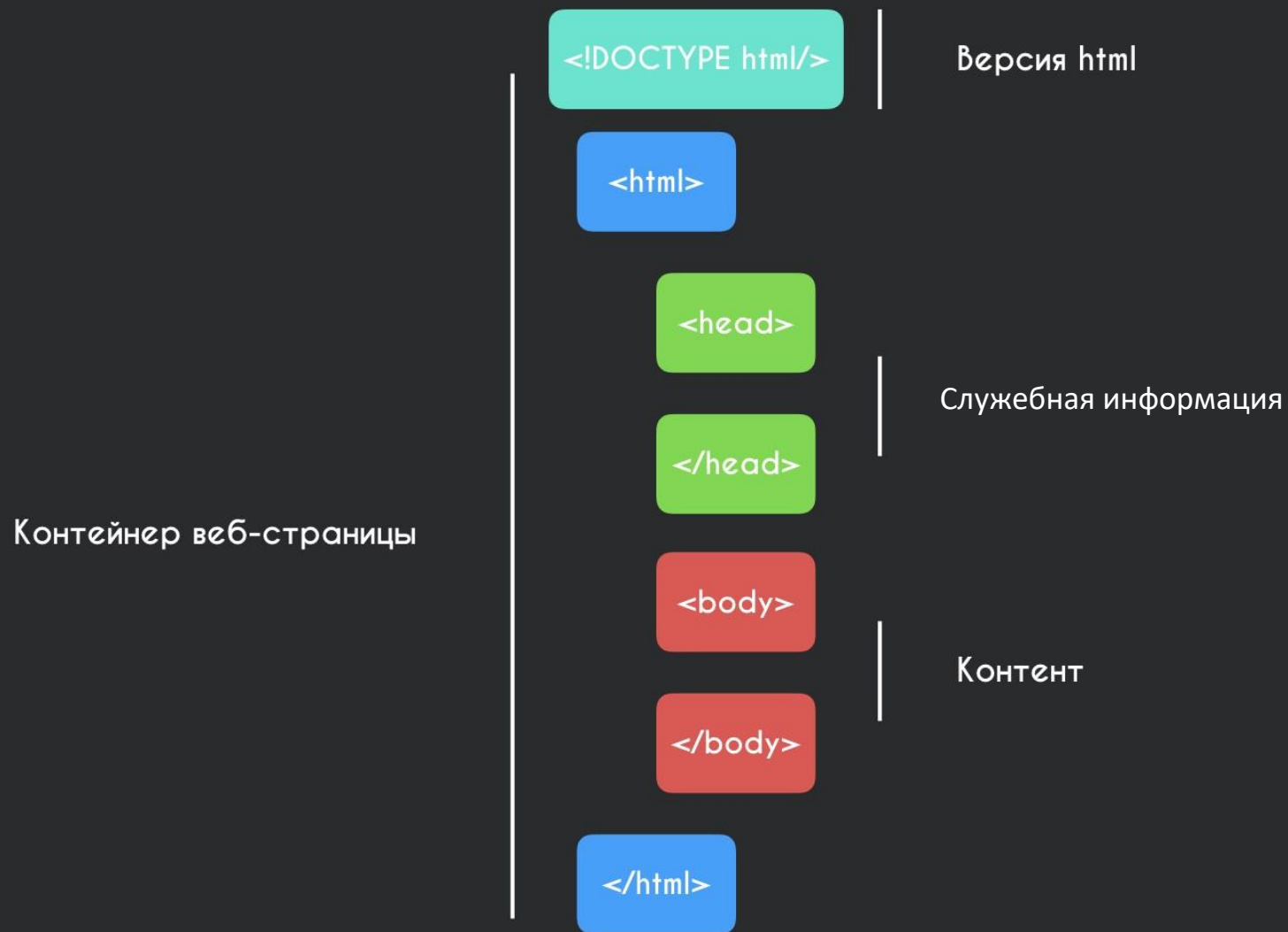
Функциональные теги: html, body, script, link, meta, и т.д.

Теги **встраиваемого контента**: img, audio, video, и т.д.

Теги **форм** (для взаимодействие с пользователем)

Теги **таблиц**

Структура HTML документа




Структура HTML документа



```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9
10 </body>
11 </html>
```

DOCTYPE

Любой HTML-документ должен начинаться с тега **DOCTYPE**, определяющему стандарт языка, по которому будет обрабатываться данный документ. Пятой версии языка HTML соответствует следующий формат:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. It displays two lines of code: line 1 is `<!DOCTYPE html>` and line 2 is empty.

```
1 <!DOCTYPE html>
2
```

Метаданные

информация о данных, находящихся в HTML-документе.

Метатеги — служебные теги в разметке HTML. Они предназначены для указания сведений поисковым роботам и браузерам.

Пример метаданных:

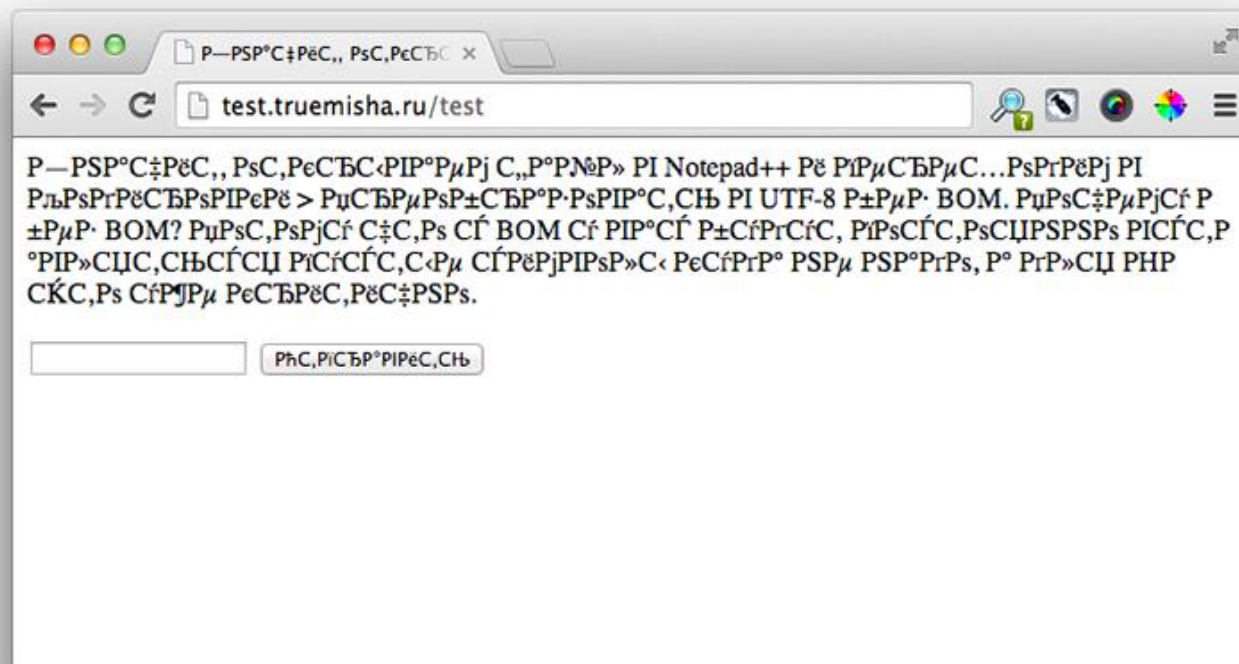
кодировка страницы, краткое описание содержимого, ключевые слова, имя автора, дата последней модификации.

В HTML метаданные HTML-документов определяются с помощью тега `<meta>`. Тег `<meta>` всегда должен располагаться в секции `head`.

Для того, чтобы указать браузеру пользователя какая кодировка используется на данной странице, необходимо использовать атрибут `charset` тега `meta`

```
<meta charset="UTF-8">
```

Проблема с кодировкой



Title

- Определяет заголовок окна браузера;
- Используется как заголовок страницы в результатах выдачи поисковых систем;
- Используется как заголовок страницы при добавлении сайта в избранное.
- Пустой заголовок, не содержащий ни одного символа, включая пробел, не допустим.
Также запрещено включать в код два и более элемента `<title>`, он должен быть только один.

Правила написания тегов:

Теги заключаются в угловые скобки `<тег>`;

Теги почти всегда парные — открывающий и закрывающий `<тег>...</тег>`.

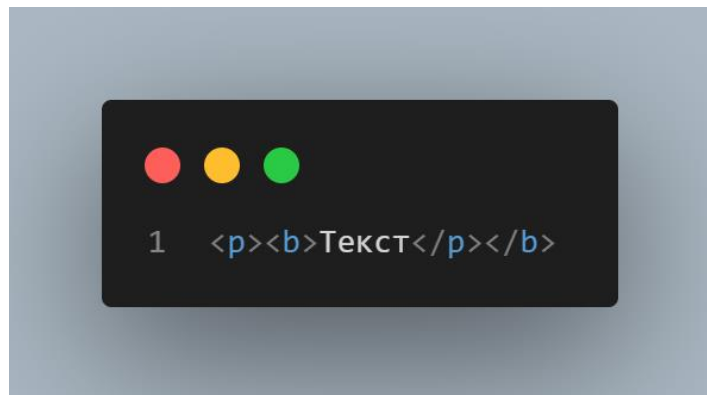
Элементы можно располагать внутри других элементов — это называется вложением

Теги закрываются зеркально `<тег1><тег2>...</тег2></тег1>`.

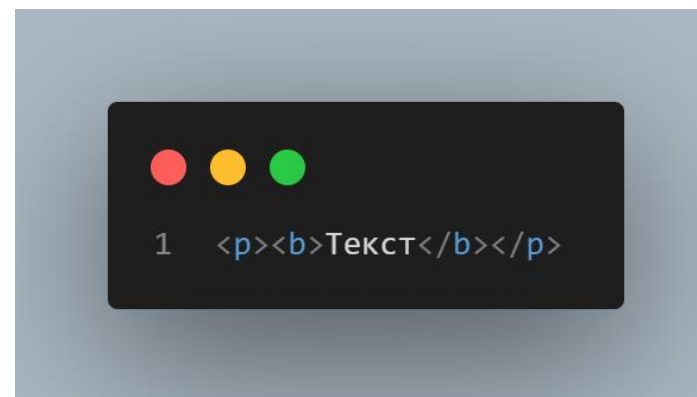
Бывают теги, которые не нужно закрывать, например, `
` или ``.

Нельзя перекрещивать теги

Не правильно!



Правильно!



`<div><h1>Заголовок</div></h1>`

`<div><h1>Заголовок</h1></div>`

Если в HTML-коде подряд идут несколько пробелов или переносов строк, то при отображении текста в браузере будет учитываться только один пробел.

Заголовки

Для создания структуры больших текстов обычно используются заголовки.

В первую очередь, заголовки должны кратко и точно описывать содержимое которое они озаглавливают. Наиболее важная информация страницы должна располагаться под заголовками большего размера, а наименее важная – под заголовками меньшего размера.

В языке HTML для выделения заголовков предусмотрено целое семейство тегов: от `<h1>` до `<h6>`. Тег `<h1>` обозначает самый важный заголовок (заголовок верхнего уровня), а тег `<h6>` обозначает подзаголовок самого нижнего уровня.

На практике редко встречаются тексты, в которых встречаются подзаголовки ниже третьего уровня. Поэтому самыми часто используемыми тегами заголовков являются: `<h1>`, `<h2>` и `<h3>`

Теги `<p></p>`

С помощью тегов `<p></p>` создаются абзацы. По умолчанию абзацы начинаются с новой строки и имеют вертикальные отступы, которыми можно управлять с помощью стилей.

`<p>`

Lorem ipsum dolor sit amet, consectetur
adipiscing elit. Facilis, veritatis?
Doloremque necessitatibus magnam
odio quas non qui dolores adipisci
dignissimos illum, veniam ex alias modi
ad. Cupiditate sapiente error aspernatur.

`</p>`

Форматирование текста

`` — жирный текст без придания важности выделенному фрагменту.

`` — расставление акцентов в тексте путём выделения его фрагментов полужирным начертанием.

`<i></i>` — выделение текста курсивом без придания важности.

`` — расставление акцентов путём выделения фрагментов текста курсивом.

`` - нижний индекс.

`` - верхний индекс

Тег `
` переводит текст на новую строку.

Тег `<hr>` добавляет в документ горизонтальную линию.

Установка цветов в HTML-документе производится с использованием таблицы стилей CSS.

Вспомните, что будет, если сделать, к примеру, два абзаца рядом - в этом случае текст, который лежит в каждом из них, будет начинаться с **НОВОЙ** строки.

Бывают, однако, ситуации, когда мы хотели бы, чтобы у нас был один абзац, но некоторый текст в нем начинался с новой строки. Зачем такое может понадобиться? К примеру, я хочу набрать стихотворение, но не хочу разбивать каждую его строку в отдельный абзац, так как это было бы не очень логично.

Чтобы сделать такое, в том месте, где должен быть разрыв строки, следует написать тег `
`. Учтите, что этот тег особенный и не имеет закрывающего.

Списки

Списком называется взаимосвязанный набор отдельных фраз или предложений.

Списки предоставляют возможность упорядочить и систематизировать разные данные и представить их в наглядном и удобном для пользователя виде.

Рассмотрим два вида списков:

- маркированный (неупорядоченный) список;
- нумерованный (упорядоченный) список;

Маркированный список

Сам список формируется с помощью элемента ``, а каждый пункт списка начинается с элемента ``

``

`Первый пункт`

`Второй пункт`

`Третий пункт`

``

- Первый пункт
- Второй пункт
- Третий пункт

Внутри `` не обязательно размещать только текст, это могут быть и другие элементы, вроде абзацев `<p>`.

Кроме того, один список `` можно вкладывать в другой, опять же, внутрь ``

```
<ul>
  <li>В
    <ul>
      <li>Первый элемент</li>
      <li>Второй элемент</li>
    </ul>
  </li>
  <li>Ж
    <ul>
      <li>Всего один элемент</li>
    </ul>
  </li>
  <li>О
    <ul>
      <li>Первый элемент</li>
      <li>Второй элемент</li>
    </ul>
  </li>
</ul>
```

- В
 - Первый элемент
 - Второй элемент
- Ж
 - Всего один элемент
- О
 - Первый элемент
 - Второй элемент

Нумерованный список

```
<ol>  
  <li>Первый элемент</li>  
  <li>Второй элемент</li>  
  <li>Третий элемент</li>  
</ol>
```

1. Первый элемент
2. Второй элемент
3. Третий элемент

Потомки и родители



```
1 <ul>
2     <li>
3         <h1>Заголовок элемента списка</h1>
4     </li>
5     <li>Элемент списка</li>
6     <li>Элемент списка</li>
7 </ul>
```

Форматирование кода

Код должен быть не только корректным с точки зрения синтаксиса языка, но и правильно оформленным для быстрого понимания структуры документа человеком, который будет читать этот код.

Для этого нужно соблюдать основное правило форматирования кода: вложенный тег на новой строке нужно начинать писать с отступом в одну табуляцию от начала тега своего родителя.

Форматировать код можно либо вручную с помощью клавиш `<Tab>` (сдвинуть текущую строку на одну табуляцию вправо) и `<Shift>+<Tab>` (сдвинуть текущую строку на одну табуляцию влево), либо с помощью функции автоформата

Комментарии

Комментарии - это такой текст, который будет проигнорирован браузером - на экране он будет не виден, но останется в исходном коде страницы.

Комментарии в HTML оформляются следующим образом: сначала уголок, знак ! и два дефиса - `<!--`, потом текст комментария, а потом два дефиса и уголок `-->`.

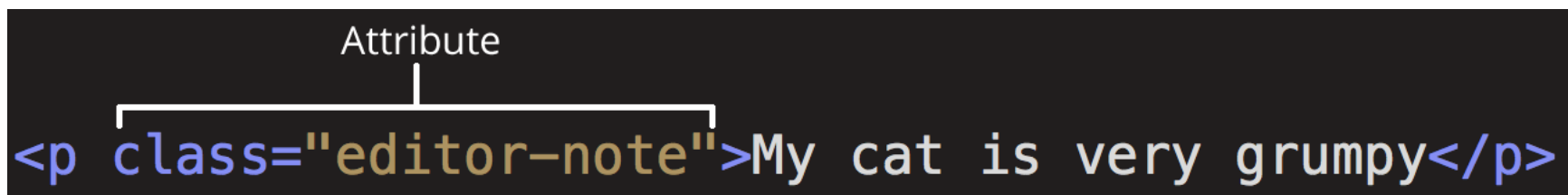
Атрибуты

В тегах также могут размещаться **атрибуты** - специальные команды, которые расширяют действие тега.

Атрибуты размещаются внутри открывающего тега в таком формате:

<тег атрибут1="значение" атрибут2="значение">

Кавычки могут быть любыми - одинарными или двойными, допустимо их вообще их не ставить, если значение атрибута состоит из одного слова (но это не желательно).



```
<p class="editor-note">My cat is very grumpy</p>
```

Есть два типа атрибутов:

атрибут со значением и логический атрибут, у которого нет значения.

- Атрибуты пишутся внутри открывающего тега,
- несколько атрибутов перечисляются через пробел,
- порядок их значения не имеет.

Атрибут со значением

Вначале пишется открывающий тег, затем через пробел имя атрибута, после чего ставится знак равно (=) и в кавычках указывается значение атрибута. Общий синтаксис такой:

```
<E атрибут="значение">...</E>
```

Здесь буквой E обозначается название произвольного элемента. Для самозакрывающих элементов всё будет аналогично, кроме содержимого и закрывающего тега.

```
<E атрибут="значение">
```

Вместо двойных кавычек ("значение") можно писать одинарные кавычки ('значение') или вообще опустить кавычки (значение). Однако хорошим тоном будет всегда писать значения атрибутов в кавычках, потому что их отсутствие может привести к неявным ошибкам.

Пример написания атрибутов

```
<p></p>
```

```
<p></p>
```

В данном примере первая строка написана правильно со всеми кавычками, а в следующей строке у атрибута alt кавычки отсутствуют.

Поскольку пробел отделяет один атрибут от другого, то браузер в качестве значения alt возьмёт только первое слово («Вид»), а слово «заголовка» будет воспринимать как новый атрибут. Но так как подобного атрибута не существует, то это приведёт к ошибке.

Каждый атрибут должен быть единственным и не должен повторяться (alt писать можно, alt alt нельзя).

Порядок атрибутов

Порядок атрибутов в элементе не имеет значения и на результат отображения элемента не влияет.

Также допустим перенос атрибутов на другую строку.

Следующие записи по своему действию равноценны.

```
  
  

```

Тег `` предназначен для внедрения графики на web страницы.

Закрывающий тег не требуется.

Адрес файла с картинкой задаётся через атрибут `src`

Атрибут `alt` - Выводит текст если картинка не загружена.

Если необходимо, то рисунок можно сделать ссылкой на другой файл, поместив `` в контейнер `<a>`.

Синтаксис: ``

Пути могут быть двух видов:

`` - относительный путь (указывается от самой страницы. Путь зависит от того где находится страница в которой мы этот путь прописываем)

`` - абсолютный путь (всегда будет указывать на картинку вне зависимости куда мы будем перемещать страницу внутри файловой системы)

Как правильно написать alt-текст

Alt — не обязательный атрибут тега ``. Он появился ещё в 1995 году, в HTML 2.0. Это альтернативное описание для изображений, которые не видят пользователи:

- из-за медленного соединения;
- из-за неправильного пути или имени файла в атрибуте `src`;
- так как пользуются скринридерами.

Когда alt-текст нужен

У любого изображения, которое иллюстрирует или дополняет текст. Например, для изображений в теге `<a>`, если у ссылки нет текстового содержимого.

```
<a href="pic_link">  
    
</a>
```

<a>

Ссылки являются теми элементами, которые делают из интернета интернет. Нажимая на ссылки, мы можем переходить с одной страницы сайта на другую. Если бы их не было - интернет был бы просто набором страниц, никак не связанных друг с другом.

Ссылка создается с помощью тега **<a>**, при этом у нее должен быть обязательный атрибут **href**, в котором хранится адрес той страницы, на которую ведет ссылка. Ссылки бывают **абсолютные** и **относительные**, кроме того, они могут вести как на ваш сайт, так и на чужой.

Универсальные атрибуты

У некоторых элементов есть свой набор характерных атрибутов, но кроме этого существуют атрибуты, которые можно добавлять к любому элементу. По этой причине они называются универсальными или глобальными атрибутами.

Перечислим лишь некоторые популярные: `class`, `id`, `lang`, `style`, `title`.

Теги `div` и `span`

У каждого из изученных нами тегов есть свой смысл. Но под все цели нельзя придумать теги.

Есть два специальных тега у которых «смысла» нет. Это теги `<div>` (сокращение от «division») и ``. Это «чистые» элементы, которые отлично подходят для визуальной группировки других элементов. Использовать эти теги рекомендуется, если более подходящих семантических тегов не нашлось.

Теги `<div>` и `` не имеют никакого оформления по умолчанию и их почти всегда используют вместе с атрибутом `class`, чтобы легко добавлять им собственные стили.

<div>

Элемент <div> (от англ. *division* — раздел, секция) является универсальным блочным элементом и предназначен для группирования элементов документа с целью изменения вида содержимого через стили. Для этого добавляется атрибут class или id с именем класса или идентификатора.

Как и при использовании других блочных элементов, содержимое <div> всегда начинается с новой строки, после него также добавляется перенос строки.

Синтаксис

```
<div>...</div>
```

Тег <div> обычно используется для группировки крупных элементов, например, нескольких абзацев, или в качестве контейнера для создания сеток страниц.

div

h1

Учитесь у лучших

div

Наши преподаватели – профессионалы, которые добились успеха в своей области. Лид-дизайнеры известных студий, маркетологи крупнейших компаний, редакторы популярных медиа: получайте опыт из первых рук, в любое время.

p

Кураторы курсов – наши лучшие выпускники, будут помогать и поддерживать вас на протяжении всего обучения.

p

Все преподаватели

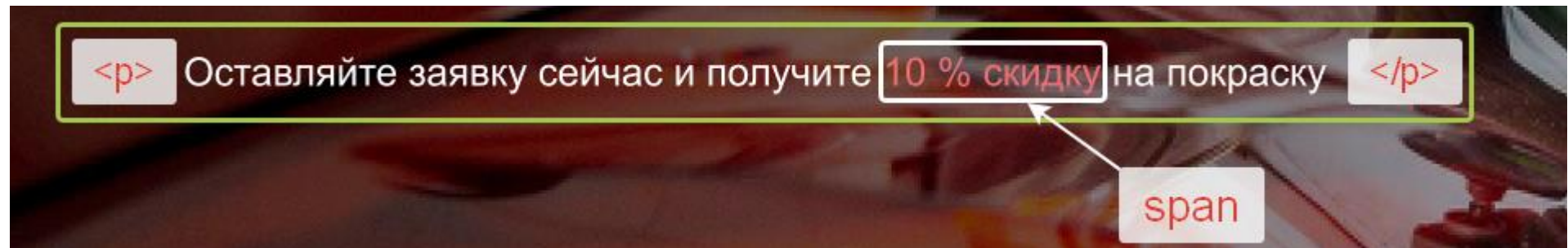
a

Универсальный строчный элемент `` (от англ. *span* — охватывать) предназначен для выделения отдельных строк, символов или других строчных элементов для дальнейшего изменения их оформления с помощью стилей. Например, внутри абзаца (`<p>`) можно изменить цвет и размер первого слова, если его выделить с помощью элемента `` и задать для него желаемый стиль.

`` используется для выделения мелких текстовых элементов: частей слов, отдельных слов или фраз, состоящих из нескольких слов:

Синтаксис

` . . . `



Атрибут `id`

Задаёт стилевой идентификатор — уникальное имя элемента, которое используется для изменения его стиля и обращения к нему через скрипты. Идентификатор в коде документа должен быть в единственном экземпляре, иными словами, встречаться только один раз.

Идентификатор состоит по меньшей мере из одного символа и не должен содержать пробел. В имя идентификатора допустимо включать цифры, символ подчёркивания, дефис и др.

Синтаксис

```
id=<имя идентификатора>
```

Атрибут `class`

Атрибут `class` задает один или несколько классов для элемента. Это делается для того, чтобы затем обратиться через CSS к группе элементов, у которых задан одинаковый класс, и применить для него определенные свойства (к примеру, сменить цвет текста, размер шрифта и так далее).

В значении допускается указывать сразу несколько классов, разделяя их между собой пробелом.

Синтаксис

```
class="<имя класса>"
```

ОФИСЫ (7)

ЗАКАЗАТЬ ЗВОНОК

ЗАПРОСИТЬ ЦЕНУ

атрибут **title**

Атрибут **target='_blank'**

атрибут **download**

Переход по ссылке внутри документа

Id и class

div, span

Devtools

A solid blue horizontal bar at the bottom of the slide.

Формы

HTML-формы — это простые элементы управления **HTML**, которые применяются для сбора информации от посетителей веб-сайта. К ним относятся текстовые поля для ввода данных с клавиатуры, списки для выбора predetermined данных, флажки для установки параметров и т. п.

Формы позволяют пользователю ввести данные, которые затем отправляются на сервер для их дальнейшей обработки и хранения или используются на стороне клиента для обновления интерфейса (например, добавление нового элемента в список или открытие и закрытие элемента интерфейса).

АВТОРИЗАЦИЯ НА САЙТЕ

Логин

.....

[Забыли пароль?](#) [ВОЙТИ](#)

<form>

- Элемент <form> (от англ. form — форма) устанавливает форму на веб-странице. Форма предназначена для обмена данными между пользователем и сервером. Область применения форм не ограничена отправкой данных на сервер, с помощью клиентских скриптов можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.
- Документ может содержать любое количество форм, но одновременно на сервер может быть отправлена только одна форма. По этой причине данные форм должны быть независимы друг от друга.
- Для отправки формы на сервер используется кнопка Submit, того же можно добиться, если нажать клавишу Enter в пределах формы. Если кнопка Submit отсутствует в форме, клавиша Enter имитирует её использование.

Атрибут тега form

```
<form action="" method="" enctype="" autocomplete=""></form>
```

`action=""` - куда пойдут данные с формы

`method=""` - каким из типов запросов пойдут эти данные

`enctype=""` - способ которым будут закодированы данные

`autocomplete=""` - указывается когда нужно включить или выключить автозаполнение полей форм. По умолчанию – on

<input>

Синтаксис <input>

<input> (от англ. *input* — ввод) является одним из разносторонних элементов формы и позволяет создавать разные части интерфейса и обеспечивать взаимодействие с пользователем. Главным образом **<input>** предназначен для создания текстовых полей, различных кнопок, переключателей и флажков.

Основной атрибут **<input>**, определяющий вид элемента — **type**.

disabled - Запрещает ввод в поле < **type="text" disabled**>

Атрибут **type** - Сообщает браузеру, к какому типу относится элемент формы.

```
<input type="button|checkbox|file|hidden|image|password|radio|reset|submit|text">
```

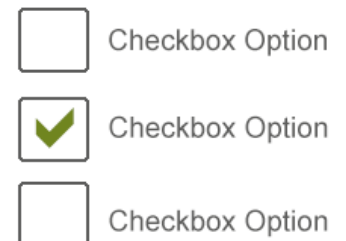
button	Кнопка.
checkbox	Флажки. Позволяют выбрать более одного варианта из предложенных.
file	Поле для отправки файла.
hidden	Скрытое поле. Оно никак не отображается на веб-странице.
image	Поле с изображением. При нажатии на рисунок данные формы отправляются на сервер.
password	Обычное текстовое поле, но отличается от него тем, что все символы показываются звездочками. Предназначено для того, чтобы никто не подглядел вводимый пароль.
radio	Переключатели. Используются, когда следует выбрать один вариант из нескольких предложенных.
reset	Кнопка для возвращения данных формы в первоначальное значение.
submit	Кнопка для отправки данных формы на сервер.
text	Текстовое поле. Предназначено для ввода символов с помощью клавиатуры.

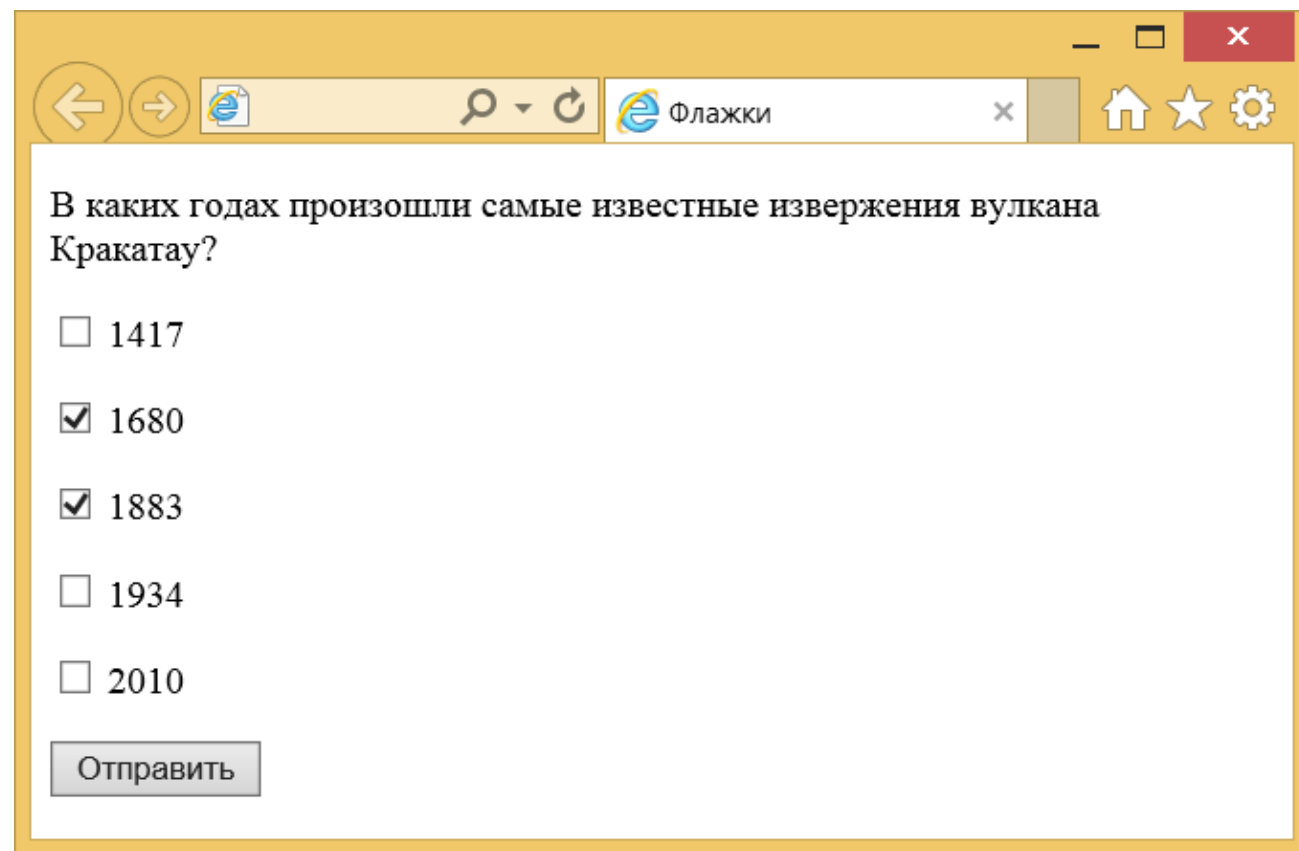
Флажки

Флажки (жарг. чекбоксы) используют, когда необходимо выбрать любое количество вариантов из предложенного списка. Если требуется выбор лишь одного варианта, то для этого следует предпочесть переключатели (radiobutton). Флажок создаётся следующим образом.

`<input type="checkbox" атрибуты>`

Атрибуты флажков	
Атрибут	Описание
checked	Предварительное выделение флажка.
name	Имя флажка для его идентификации обработчиком формы.
value	Задаёт, какое значение будет отправлено на сервер.





В каких годах произошли самые известные извержения вулкана Кракатау?

- ☐ 1417
- ☒ 1680
- ☒ 1883
- ☐ 1934
- ☐ 2010

Отправить

The image shows a web browser window with a yellow title bar. The address bar contains the text 'Флажки' and a search icon. The main content area displays a quiz question in Russian: 'В каких годах произошли самые известные извержения вулкана Кракатау?'. Below the question are five radio button options: 1417, 1680, 1883, 1934, and 2010. The options 1680 and 1883 are selected, indicated by checkmarks inside the boxes. At the bottom left of the content area is a button labeled 'Отправить' (Send).

Пример

```
✓ <form action="handler.php">
  <p>В каких годах произошли самые известные извержения вулкана Кракатау?</p>
  <p><input type="checkbox" name="a" value="1417"> 1417</p>
  <p><input type="checkbox" name="a" value="1680" checked> 1680</p>
  <p><input type="checkbox" name="a" value="1883" checked> 1883</p>
  <p><input type="checkbox" name="a" value="1934"> 1934</p>
  <p><input type="checkbox" name="a" value="2010"> 2010</p>
  <p><input type="submit" value="Отправить"></p>
</form>
```

Переключатели

Переключатели (жарг. радиокнопки) используют, когда необходимо выбрать один единственный вариант из нескольких предложенных. Создаются следующим образом.

```
<input type="radio">
```

Атрибуты переключателей

Атрибут	Описание
name	Имя группы переключателей для идентификации поля. Поскольку переключатели являются групповыми элементами, то имя у всех элементов группы должно быть одинаковым.
disabled	Блокирует переключатель для выбора.
form	Идентификатор формы для связывания кнопки с элементом <code><form></code> .
type	Для переключателя значение должно быть <code>radio</code> .
checked	Предварительное выделение переключателя. По определению, набор переключателей может иметь только один выделенный пункт, поэтому добавление <code>checked</code> сразу к нескольким полям не даст особого результата. В любом случае, будет отмечен элемент, находящийся в коде HTML последним.
value	Задаёт, какой текст будет отправлен на сервер. Здесь уже каждый элемент должен иметь свое уникальное значение, чтобы можно было идентифицировать, какой пункт был выбран пользователем.
autofocus	Переключатель получает фокус после загрузки документа.
required	Выбор переключателя перед отправкой формы становится обязательным.

Атрибут placeholder

Выводит текст внутри поля формы, который исчезает при получении фокуса или при наборе текста. Обычно отображается серым цветом.

```
<input placeholder="<текст>">
```

```
<form action="handler.php">
  <p><input type="search" placeholder="Введите текст для поиска">
    <input type="submit" value="Искать"></p>
</form>
```


Атрибут required

Устанавливает поле формы обязательным для заполнения перед отправкой формы на сервер. Если обязательное поле пустое, браузер выведет сообщение об ошибке, а форма отправлена не будет. Вид и содержание сообщения зависит от браузера и меняться пользователем не может.

`<input required>`

```
<form action="">
  <p><input name="user" required placeholder="Ваше имя"></p>
  <p><input type="submit" value="Отправить"></p>
</form>
```



Заполните это поле.

<select>

<select> (от англ. *selection* — выбор) позволяет создать элемент интерфейса в виде раскрывающегося списка, а также список с одним или множественным выбором.

Каждый пункт создаётся с помощью элемента <option>, который должен быть вложен в контейнер <select>. Если планируется отправлять данные списка на сервер, то требуется поместить <select> внутри формы. Это также необходимо, когда к данным списка идёт обращение через скрипты.

```
<select>
  <option>Пункт 1</option>
  <option>Пункт 2</option>
</select>
```

Атрибуты

<u>disabled</u>	Блокирует доступ и изменение элемента.
<u>multiple</u>	Позволяет одновременно выбирать сразу несколько элементов списка.
<u>name</u>	Имя элемента для отправки на сервер или обращения через скрипты.
<u>required</u>	Список обязателен для выбора перед отправкой формы.
<u>size</u>	Количество отображаемых строк списка.

<textarea>

Синтаксис

`<textarea></textarea>`

`<textarea>` представляет собой элемент формы для создания области, в которую можно вводить несколько строк текста. В отличие от элемента `<input>` в текстовом поле допустимо делать переносы строк, они сохраняются при отправке данных на сервер.

Введите ваш отзыв:

Отправить

<button>

```
<button>...</button>
```

Элемент `<button>` (от англ. *button* — кнопка) создаёт на веб-странице кнопки и по своему действию напоминает результат, получаемый с помощью `<input>` (с атрибутом `type="button | reset | submit"`). В отличие от этого элемента, `<button>` предлагает расширенные возможности по созданию кнопок. Например, на подобной кнопке можно размещать любые элементы HTML, в том числе изображения. Используя стили можно определить вид кнопки путём изменения шрифта, цвета фона, размеров и других параметров.

```
<button type="button | reset | submit">...</button>
```

button	Обычная кнопка. Удобно использовать для создания собственных кнопок — вы можете определить их поведение через JavaScript.
reset	Кнопка для очистки введённых данных формы и возвращения значений в первоначальное состояние.
submit	Кнопка для отправки данных формы на сервер.

Пример

```
<form action="">  
  <p><input type="text" name="user"></p>  
  <p><button type="reset">Очистить форму</button>  
    <button type="submit">Отправить форму</button></p>  
</form>
```

<label>

устанавливает связь между определённой меткой, в качестве которой обычно выступает текст, и элементом формы ([<input>](#), [<select>](#), [<textarea>](#)). Такая связь необходима, чтобы изменять значения элементов формы при щелчке курсором мыши на текст.

Существует два способа связывания объекта и метки.

Первый заключается в использовании идентификатора [id](#) внутри элемента формы и указании его имени в качестве атрибута [for](#) элемента `<label>`.



```
1 <input id="<идентификатор>">  
2   <label for="<идентификатор>">Текст</label>
```

При втором способе элемент формы помещается внутрь контейнера `<label>`.



```
1   <label><input> Текст</label>  
2
```

Таблицы

Структура таблицы

<thead></thead>

<tbody></tbody>

<tfoot> </tfoot>

`<table> </table>`

служит контейнером для элементов, определяющих содержимое таблицы.

Любая таблица состоит из строк и ячеек.

Элемент `<caption> </caption>` создает заголовок таблицы.

Элемент `<tr> </tr>` служит для создания строки таблицы.

Элемент `<td> </td>` служит для создания ячейки таблицы. должен размещаться внутри контейнера `<tr>`, который в свою очередь располагается внутри `<table>`.

Элемент `<th> </th>` служит для создания заголовков для столбца таблицы.

<caption>...</caption>

(от англ. *caption* — заголовок) предназначен для создания заголовка к таблице и может размещаться только внутри контейнера **<table>**, причём сразу после открывающего тега. Такой заголовок представляет собой текст, по умолчанию отображаемый перед таблицей и описывающий её содержание.

<th>

Элемент `<th>` (от англ. *table header cell* — ячейка заголовка таблицы) предназначен для создания одной ячейки таблицы, которая обозначается как заголовочная. Текст в такой ячейке отображается браузером обычно жирным шрифтом и выравнивается по центру. Элемент `<th>` должен размещаться внутри контейнера `<tr>`, который в свою очередь располагается внутри `<table>`.

<table>

<td></td>	<td></td>	<td></td>	</tr>
<td></td>	<td></td>	<td></td>	</tr>
<td></td>	<td></td>	<td></td>	</tr>
<td></td>	<td></td>	<td></td>	</tr>

</table>



```
1 <table border="1">
2     <caption>Название таблицы</caption>
3     <tr>
4         <th>Заглавная ячейка 1</th>
5         <th>Заглавная ячейка 2</th>
6         <th>Заглавная ячейка 3</th>
7     </tr>
8     <tr>
9         <td>ячейка 1</td>
10        <td>ячейка 2</td>
11        <td>ячейка 3</td>
12    </tr>
13 </table>
```

Название таблицы

Заглавная ячейка 1	Заглавная ячейка 2	Заглавная ячейка 3
ячейка 1	ячейка 2	ячейка 3

<tbody> ... </tbody>

Элемент <tbody> (от англ. *table body* — тело таблицы) предназначен для хранения одной или нескольких строк таблицы. Это позволяет создавать структурные блоки, к которым можно применять единое оформление через стили, а также управлять их видом через скрипты.

<thead> ... </thead>

(от англ. *table head* — голова или шапка таблицы) предназначен для хранения одной или нескольких строк, которые представлены вверху таблицы. Допустимо использовать не более одного элемента **<thead>** в пределах одной таблицы, и он должен идти в исходном коде сразу после открывающего тега **<table>** или **<caption>** (если он есть).

<tfoot> ... </tfoot>

(от англ. *table foot* — подвал таблицы) представляет собой «подвал» таблицы и отображается внизу таблицы. Предназначен для информации о колонках таблицы.

Разделы <thead>, <tfoot> и <tbody> должны содержать одинаковое число колонок и по меньшей мере одну строку, которая определяется через элемент <tr>.

В таблице разрешается использовать только один элемент <tfoot>.

Синтаксис

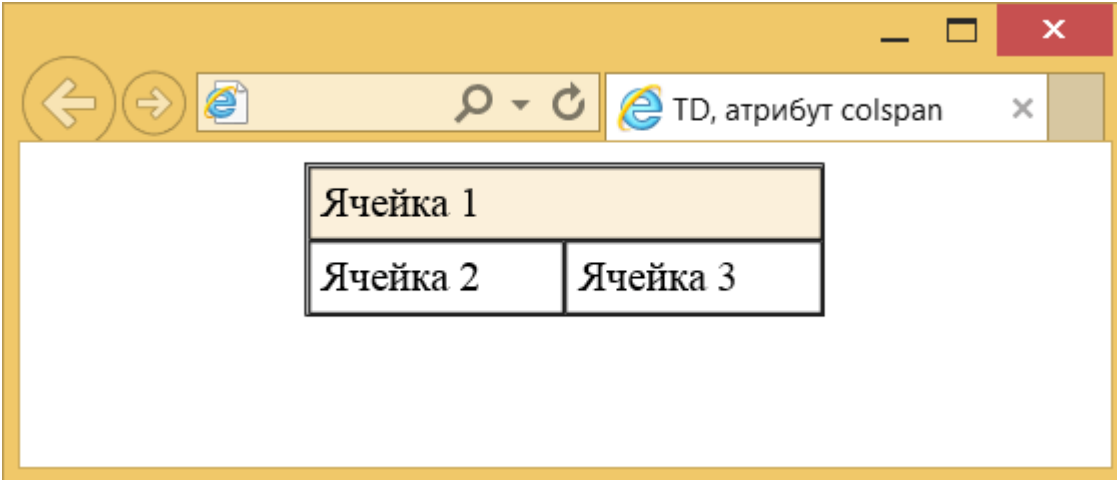
```
<table>
  <thead>...</thead>
  <tbody>...</tbody>
  <tfoot>...</tfoot>
</table>
```

<thead> </thead>						
	№ п/п	Наименование товара	Ед. изм.	Количество	Цена за ед. изм., руб.	Стоимость, руб.
<tbody> </tbody>	1.	Томаты свежие	кг	15,20	69,00	1048,80
	2.	Огурцы свежие	кг	2,50	48,00	120,00
<tfoot> </tfoot>	ИТОГО:					1168,80

Атрибут colspan

Устанавливает число ячеек, которые должны быть объединены по горизонтали. Этот атрибут имеет смысл для таблиц, состоящих из нескольких строк.

```
<td colspan="число">...</td>
```

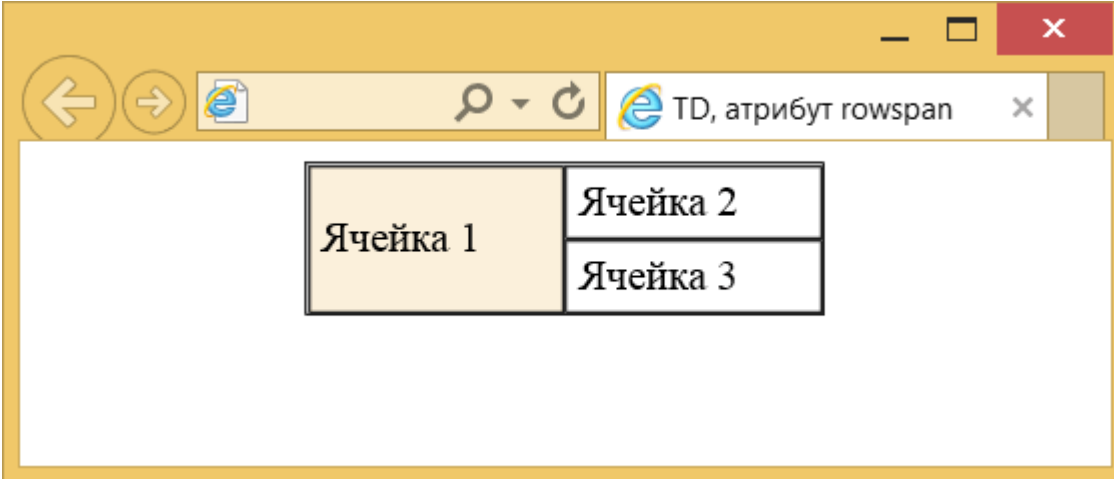


Ячейка 1	
Ячейка 2	Ячейка 3

Атрибут rowspan

Устанавливает число ячеек, которые должны быть объединены по вертикали. Этот атрибут имеет смысл для таблиц, состоящих из нескольких строк.

```
<td rowspan="число">...</td>
```



The screenshot shows a web browser window with a yellow title bar and a single tab titled "TD, атрибут rowspan". The browser's address bar and navigation buttons are visible. The main content area displays a table with three cells. The first cell, labeled "Ячейка 1", is highlighted in light yellow and spans two rows. The second and third cells are labeled "Ячейка 2" and "Ячейка 3" respectively, and are in the same column as "Ячейка 1".

Ячейка 1	Ячейка 2
	Ячейка 3

Пример

Автомобили

Номер п/п	Марка	Модель	Стоимость руб.
1	Тойота	Corolla	1900000
2	Тойота	Camry	2400000
3	Тойота	RAV 4	2600000
Итого			6900000