

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.  
Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Единицы измерения в HTML. Физическое и логическое форматирование текста  
HTML-документа. Ссылки. Использование графики. Списки. Таблицы.  
Внедрение аудио и видео. Геолокация и карты

## Единицы измерения в HTML

**Размер шрифта** – это некоторая «условная единица», которая встроена в шрифт.

Она обычно чуть больше, чем расстояние от верха самой большой буквы до низа самой маленькой. То есть, предполагается, что в эту высоту помещается любая буква или их сочетание. Но при этом «хвосты» букв, таких как *p*, *g* могут заходить за это значение, то есть вылезать снизу. Поэтому обычно высоту строки делают чуть больше, чем размер шрифта.

Для задания размеров различных элементов, в CSS используются абсолютные и относительные единицы измерения. Абсолютные единицы не зависят от устройства вывода, а относительные единицы определяют размер элемента относительно значения другого размера.

**Относительные** единицы обычно используют для работы с текстом, либо когда надо вычислить процентное соотношение между элементами. Основные относительные единицы.

Единица	Описание
Em	Высота шрифта текущего элемента
Ex	Высота символа <i>x</i>
Px	Пиксел
%	Процент

Изменяемое значение, которое зависит от размера шрифта текущего элемента (он устанавливается через стилевое свойство *font-size*).

**1 em** – текущий размер шрифта.

Можно брать любые пропорции от текущего шрифта: 2em, 0.5em и т.п.

В каждом браузере заложен размер текста, применяемый в том случае, когда этот размер явно не задан. Поэтому изначально **1em** равен размеру шрифта, заданного в браузере по умолчанию. Соответственно, устанавливая размер текста для всей страницы в **em**, мы работаем именно с этим параметром. В том случае, когда **em** используется для определенного элемента, за **1em** принимается размер шрифта его родителя.

```
<div style="font-size:1.5em">  
  Страусы  
  <div style="font-size:1.5em">Живут также в Африке</div>  
</div>
```

**ex** определяется как высота символа «*x*» в нижнем регистре. На **ex** распространяются те же правила, что и для **em**, а именно, он привязан к размеру шрифта, заданного в браузере по умолчанию, или к размеру шрифта родительского элемента.

В спецификации указаны единицы **ex** и **ch**, которые означают размер символа «*x*» и размер символа «*0*». Эти размеры присутствуют в шрифте всегда, даже если по коду этих символов в шрифте находятся другие значения, а не именно буква «*x*» и ноль «*0*». В этом случае они носят более условный характер. Эти единицы используются чрезвычайно редко, так как «размер шрифта» **em** обычно вполне подходит.

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.  
Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

**Пиксель px** – это самая базовая, абсолютная и окончательная единица измерения, отображаемая монитором или другим подобным устройством, например, смартфоном. Размер пиксела зависит от разрешения устройства и его технических характеристик.

Количество пикселей задаётся в настройках разрешения экрана, один **px** – это как раз один такой пиксель на экране. Все значения браузер в итоге пересчитает в пиксели.

Пиксели могут быть дробными, например размер можно задать в **16.5px**. Это совершенно нормально, браузер сам использует дробные пиксели для внутренних вычислений. К примеру, есть элемент шириной в **100px**, его нужно разделить на три части – волей-неволей появляются **33.333...px**. При окончательном отображении дробные пиксели, конечно же, округляются и становятся целыми.

Для мобильных устройств, у которых много пикселей на экране, но сам экран маленький, чтобы обеспечить читаемость, браузер автоматически применяет масштабирование.

Достоинства **px**: чёткость и понятность

Недостатки **px**: другие единицы измерения – в некотором смысле «мощнее», они являются относительными и позволяют устанавливать соотношения между различными размерами.

```
<div style="font-size:24px">  
  Страусы  
  <div style="font-size:24px">Живут также в Африке</div>  
</div>
```

Проценты %, как и **em** – относительные единицы. Как правило, процент будет от значения свойства родителя с тем же названием, но не всегда. Это очень важная особенность процентов, про которую, увы, часто забывают.

Вот пример с %, он выглядит в точности так же, как с **em**:

```
<div style="font-size:150%">  
  Страусы  
  <div style="font-size:150%">Живут также в Африке</div>  
</div>
```

В примере выше процент берётся от размера шрифта родителя.

А вот примеры-исключения, в которых % берётся не так:

При установке свойства **margin-left** в %, процент берётся от ширины родительского блока, а вовсе не от его **margin-left**.

При установке свойства **line-height** в %, процент берётся от текущего размера шрифта, а вовсе не от **line-height** родителя.

Для **width/height** обычно процент от ширины/высоты родителя, но при **position:fixed**, процент берётся от ширины/высоты окна (а не родителя и не документа).

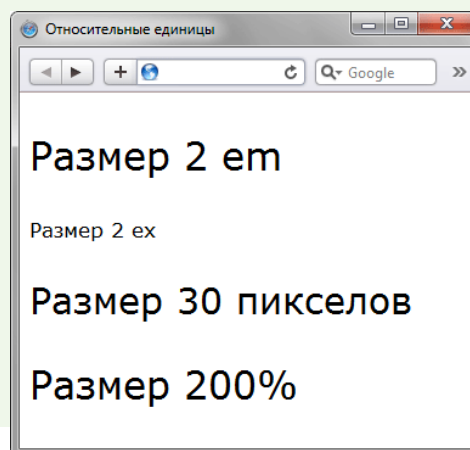
Пример: использование относительных единиц

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">  
<html>  
  <head>  
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  
    <title>Относительные единицы</title>  
    <style type="text/css">  
      .em, .ex, .px, .percent {  
        font-family: Verdana, Arial, sans-serif;
```

```

}
.em { font-size: 2em; }
.ex { font-size: 2ex; }
.px { font-size: 30px; }
.percent { font-size: 200%; }
</style>
</head>
<body>
<p class="em">Размер 2 em</p>
<p class="ex">Размер 2 ex</p>
<p class="px">Размер 30 пикселей</p>
<p class="percent">Размер 200%</p>
</body>
</html>

```



Абсолютные единицы применяются реже, чем относительные и, как правило, при работе с текстом.

Единица	Описание
In	Дюйм (1 дюйм равен 2,54 см)
Cm	Сантиметр
Mm	Миллиметр
Pt	Пункт (1 пункт равен 1/72 дюйма)
Pc	Пика (1 пика равна 12 пунктам)

Существуют также «производные» от пикселя единицы измерения: *mm*, *cm*, *pt* и *pc*, но они давно отправились на свалку истории.

Самой, пожалуй, распространенной единицей является пункт, который используется для указания размера шрифта. Многие люди привыкли задавать размер шрифта в текстовых редакторах, например, 12. А что это число означает, не понимают. Так это именно пункты и есть, они родные. Конечно они нам не родные, мы привыкли измерять все в миллиметрах и подобных единицах, но пункт, пожалуй, единственная величина из не метрической системы измерения, которая используется у нас повсеместно. И все благодаря текстовым редакторам и издательским системам.

```

1mm (мм) = 3.8px
1cm (см) = 38px
1pt (типографский пункт) = 4/3 px
1pc (типографская пика) = 16px

```

Так как браузер пересчитывает эти значения в пиксели, то смысла в их употреблении нет.

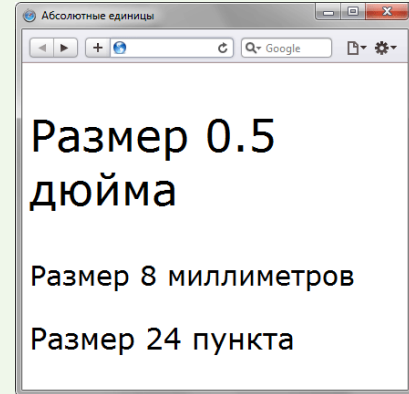
Почему в сантиметре **см** содержится ровно **38 пикселей**?

В реальной жизни сантиметр – это эталон длины, одна сотая метра. А пиксель может быть разным, в зависимости от экрана. Но в формулах выше под пикселем понимается «сферический пиксель в вакууме», точка на «стандартизованном экране», характеристики которого описаны в спецификации. Поэтому ни о каком соответствии **см** реальному сантиметру здесь нет и речи. Это полностью синтетическая и производная единица измерения, которая не нужна.

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Пример: использование абсолютных единиц

```
<!DOCTYPE HTML>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Абсолютные единицы</title>
  <style type="text/css">
    .in, .mm, .pt {
      font-family: Verdana, Arial, sans-serif;
    }
    .in { font-size: 0.5in; }
    .mm { font-size: 8mm; }
    .pt { font-size: 24pt; }
  </style>
</head>
<body>
  <p class="in">Размер 0.5 дюйма</p>
  <p class="mm">Размер 8 миллиметров</p>
  <p class="pt">Размер 24 пункта</p>
</body>
</html>
```



Результат использования абсолютных единиц измерения показан ниже.

Единица **rem**: смесь **px** и **em**

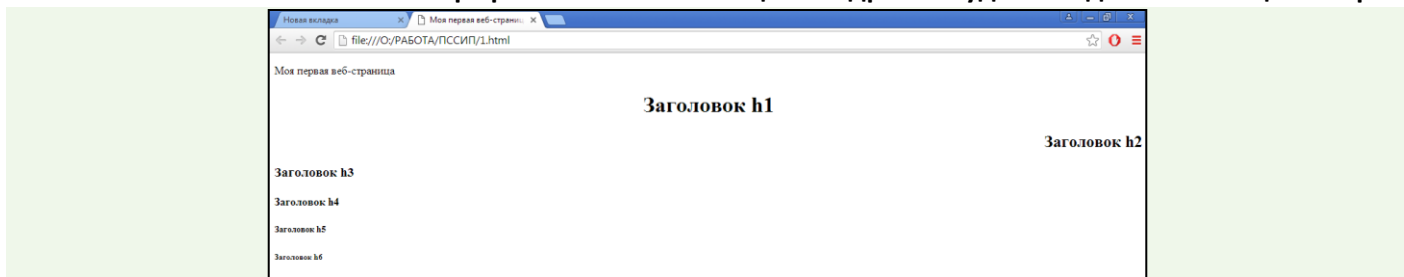
Относительно экрана: **vw**, **vh**, **vmin**, **vmax**. Эти значения были созданы, в первую очередь, для поддержки мобильных устройств.

## Физическое и логическое форматирование текста HTML-документа

HTML не задает конкретные и точные атрибуты форматирования документа. Конкретный вид документа окончательно определяет только браузер на компьютере пользователя Интернета.

Пример использования тегов:

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;
  charset=windows-1251">
  <title>Моя первая веб-страница</title>
</head>
<body>
  <p align="left">Моя первая веб-страница</p>
  <h1 align="center">Заголовок h1</h1>
  <h2 align="right">Заголовок h2</h2>
  <h3 align="justify">Заголовок h3</h3>
  <h4>Заголовок h4</h4>
  <h5>Заголовок h5</h5>
  <h6>Заголовок h6</h6>
</body>
</html>
```



**Форматирование текста** — средства его изменения, такие как выбор начертания шрифта и использование эффектов, позволяющих менять вид текста.

В предыдущих версиях HTML (до HTML5) элементы, отвечающие за форматирование текста, делились на две группы:

- теги логического форматирования;
- теги физического форматирования.

*Однако в пятой версии языка все элементы физической разметки, которые не были удалены, изменили свое назначение и стали относиться к группе элементов логического форматирования.*

**Физическое форматирование 'html'-документа** — это процесс форматирования 'html'-кода при помощи набора соответствующих элементов разметки, которые используются в основном для определения внешнего вида своего содержимого при отображении его браузером.

Приведем описание тегов, которые в предыдущих версиях HTML можно было смело отнести к тегам физического форматирования:

**<b>** (от англ. bold) — элемент разметки 'b', сформированный данным тегом, отображает свое содержимое полужирным шрифтом; в HTML 5 он обозначает стилистическое усиление своего содержимого, например, ключевых слов, которые выделяются в типографике полужирным начертанием;

**<i>** (от англ. italic) — элемент разметки 'i', сформированный данным тегом, отображает свое содержимое курсивом; в HTML 5 он обозначает дополнительное выделение своего содержимого, например, иностранных слов, технических терминов, вставок рукописного текста, короче того, что выделяется курсивом в типографике;

**<u>** (от англ. underline) — содержимое элемента разметки 'u', сформированного данным тегом, отображается подчеркнутым; в HTML 5 он в основном используется для стилистического выделения слов с орфографическими ошибками или имен собственных в китайском языке;

**<s>** (от англ. strike out) — содержимое элемента разметки 's', сформированного данным тегом, отображается зачеркнутым; в HTML 5 он используется для текста, который потерял свою актуальность, например, для старой цены продукта;

**<sub>** (от англ. subscript) — элемент разметки 'sub', сформированный данным тегом, отображает свое содержимое в виде нижнего индекса;

**<sup>** (от англ. superscript) — элемент разметки 'sup', сформированный данным тегом, отображает свое содержимое в виде верхнего индекса.

*Все перечисленные элементы формируются соответствующими парными тегами и отображаются браузерами как строчные элементы.*

Пример:

```
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>Элементы физической разметки</title>
</head>
<body>
  <b>Я &ndash; просто жирный текст.</b> <br>
  <i>Я &ndash; курсивный.</i> <br>
  <u>Я &ndash; подчеркнутый.</u> <br>
  <s>Я &ndash; зачеркнутый.</s> <br>
  А вот это выражение похоже на странную формулу:<br>
  a<sup>верхний индекс</sup> + b<sub>нижний индекс</sub>.<br><br>
  Для изменения внешнего вида используйте не нас, а стили CSS!
</body>
</html>
```

Как видим, указанные теги вполне можно было бы использовать для физической разметки документа. Тем более, что сами названия явно указывают на предназначение элементов: *italic*, *bold*, *underline* и т.д. Однако все это пережитки старых версий, и применять какие-либо теги только лишь для изменения внешнего вида текста крайне не рекомендуется. Ведь в HTML 5 все теги имеют логическую нагрузку, хотя отношение некоторых из них к логическим и выглядит несколько натянутым. Однако в любом случае, теперь HTML 5 практически полностью используется в веб-программировании, как инструмент логической разметки документа, а за внешний вид отвечает 'CSS'.

**Логическое форматирование 'html'-документа** – это процесс форматирования 'html'-кода при помощи набора соответствующих элементов разметки, которые предназначены главным образом для структурной, логической разметки документа, определяя степень важности своего содержимого и его отношение к тому или иному типу данных, а также выделяя смысловое отличие своего содержимого от окружающего контекста.

Приведем описание тегов логического форматирования:

**<em>** (от англ. emphasis) – элемент разметки 'em', сформированный данным тегом, предназначен для акцентирования текста (обращает внимание на его важность) и отображает свое содержимое курсивом;

**<strong>** – элемент разметки 'strong', сформированный данным тегом, предназначен для еще большего акцентирования текста (делает его еще более важным) и отображает свое содержимое полужирным шрифтом;

**<cite>** – элемент разметки 'cite', сформированный данным тегом, предназначен для выделения сносок на другой материал и отображает свое содержимое курсивом;

**<code>** – элемент разметки 'code', сформированный данным тегом, предназначен для выделения текста программного кода и отображает свое содержимое моноширинным шрифтом;

**<kbd>** (от англ. keyboard) – элемент разметки 'kbd', сформированный данным тегом, предназначен для выделения текста, который должен быть введен с клавиатуры или используется для названия клавиш клавиатуры; элемент отображает свое содержимое моноширинным шрифтом;

**<var>** (от англ. variable) – элемент разметки 'var', сформированный данным тегом, предназначен для выделения переменных компьютерных программ и отображает свое содержимое курсивом;

**<samp>** (от англ. sample) – элемент разметки 'samp', сформированный данным тегом, предназначен для выделения текста, который является результатом вывода компьютерной программы, и отображает свое содержимое моноширинным шрифтом;



Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.

Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

**<dfn>** (от англ. definition) – элемент разметки 'dfn', сформированный данным тегом, предназначен для выделения терминов, которые встречаются в тексте впервые, и отображает свое содержимое курсивом;

**<abbr>** (от англ. abbreviation) – элемент разметки 'abbr', сформированный данным тегом, предназначен для выделения аббревиатур и обычно используется с атрибутом title, содержащим расшифровку аббревиатуры; текст данного элемента браузерами никак не выделяется, сохраняя исходное форматирование;

**<q>** (от англ. quote) – элемент разметки 'q', сформированный данным тегом, предназначен для выделения в тексте небольших цитат и отображает свое содержимое в кавычках;

**<ins>** (от англ. inserted) – элемент разметки 'ins', сформированный данным тегом, предназначен для выделения текста, который был добавлен в новую версию документа, и отображает свое содержимое подчеркнутым;

**<del>** (от англ. deleted) – элемент разметки 'del', сформированный данным тегом, предназначен для выделения текста, который был удален в новой версии документа, и отображает свое содержимое зачеркнутым;

**<small>** – элемент разметки 'small', сформированный данным тегом, предназначен для выделения текста, который можно отнести к надписям мелким шрифтом или пометкам, как, например, второстепенная информация в конце юридических документов об отказе от ответственности или же информации о лицензии; элемент отображает свое содержимое моноширинным шрифтом.

Все перечисленные элементы также формируются соответствующими парными тегами и отображаются браузерами как строчные элементы.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Элементы логической разметки</title>
</head>
<body>
<p>
  <dfn style="color: blue">PHP</dfn> &ndash; скриптовый язык общего назначения.
  <br> Узнайте, как расшифровывается аббревиатура
  <em style="color: blue"><abbr title="Hypertext Preprocessor">PHP</abbr></em>,
  наведя на нее курсор. <br>
  Хотя первоначально аббревиатура <em style="color: blue">PHP</em> означала
  <em><del>Personal Home Page Tools</del></em>, <br>теперь ее стоит
  расшифровывать как <strong><ins>Hypertext Preprocessor</ins></strong>. <br>
</p>
<p>
  Давайте посмотрим на фрагмент кода <em style="color: blue">PHP</em>:
  <code>echo "Всем привет от PHP!";</code>. <br>
  Как сказал автор кода на своей страничке <cite>http://www.fdpdpdf.com</cite>:
  <br> <q style="color: red">Великий код!</q>. <br>
</p>
<p>
  Для запуска кода нажмите на клавиатуре <kbd>'ALT'+ 'SHIFT'+ 'Поехали! '</kbd>. <br>
  На экран будет выведено приветствие: "<samp>Всем привет от PHP!</samp>". <br>
  Убедились? Оператор <var style="color: green">echo</var> &ndash; это сила! <br>
</p>
  <small>Все права на пример защищены марсианским законодательством.</small>
</body>
</html>
```

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Не смотря на то, что при логической разметке внешний вид формируемого текста по умолчанию не всегда устраивает разработчика, изменить его не составляет труда при помощи стилей CSS. При этом логика разметки никоим образом не страдает, и для поисковых машин такие изменения не будут считаться решающими.

В завершение пункта отдельно приведем описание еще двух элементов логической разметки документа:

**<address>** – элемент разметки 'address', сформированный данным парным тегом, предназначен для выделения информации об авторе, например, ссылка на его ресурс, адрес и т.д.; элемент отображает свое содержимое курсивом, а браузеры отображают как блочный элемент.

**<blockquote>** – элемент разметки 'blockquote', сформированный данным парным тегом, предназначен для выделения длинных цитат, в отличие от элемента 'q', при помощи которого выделяются небольшие цитаты; браузеры отображают элемент как блочный.

```
<html>
<head>
  <meta charset="utf-8">
  <title>Элементы 'address' и 'blockquote'</title>
</head>
<body>
  <h3>Бернард Шоу</h3>

  <blockquote>
    Мир настолько сгнил, что даже влюбиться в кого-то – <br>
    это самый большой риск, который мы можем себе позволить. <br>
    Нас сжимает изнутри от вероятности, что это окажется не <br>
    взаимно или агрессивно воспринято. Люди разучились любить, <br>
    миром правят потребительские отношения.
  </blockquote>

  <address>Связаться с автором цитаты не получится.</address>
</body>
</html>
```

Теги форматирования могут быть вложенными друг в друга. При этом нужно внимательно следить, чтобы один контейнер находился целиком в другом контейнере.

Практически ко всем тегам применим атрибут **title** - всплывающая подсказка, поэтому, если в тексте необходимо выделить некую аббревиатуру, желательно давать к ней расшифровку, используя данный атрибут.

Пример:

```
<abbr title="Cascading Style Sheets">CSS</abbr> позволит Вам без труда изменить стиль
любого тега логического форматирования текста!
<acronym title="коммунистический союз молодежи">комсомол</acronym>
```

Кроме того, теги логического форматирования можно переопределить с использованием CSS.

**HTML теги** (имя тега, описание, значение свойства display)

Текст		
<h1></h1> - <h6></h6>	заголовки шести уровней	block



**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

<p></p>	параграфы в тексте	block
 	перенос текста на новую строку	none
<hr>	горизонтальная линия	block
<wbr>	возможное место разрыва длинной строки	none
<blockquote></blockquote>	большая цитата	block
<cite></cite>	источник цитирования	inline
<q></q>	краткая цитата	inline
<code></code>	фрагмент программного кода	inline
<kbd></kbd>	текст, вводимый пользователем с клавиатуры	inline
<pre></pre>	выводит текст с пробелами и переносами	block
<samp></samp>	результат выполнения сценария	inline
<var></var>	выделяет переменные из программ	inline
<del></del>	перечёркивает текст, помечая как удаленный	inline
<s></s>	перечёркивает неактуальный текст	inline
<dfn></dfn>	выделяет термин курсивом	inline
<em></em>	выделяет важные фрагменты текста курсивом	inline
<i></i>	выделяет текст курсивом без акцента	inline
<strong></strong>	выделяет полужирным важный текст	inline
<b></b>	задает полужирное начертание отрывка текста, без дополнительного акцента	inline
<ins></ins>	подчёркивает изменения в тексте	inline
<u></u>	выделяет отрывок текста подчёркиванием, без дополнительного акцента	inline
<mark></mark>	выделяет фрагменты текста желтым фоном	inline
<small></small>	отображает текст шрифтом меньшего размера	inline
<sub></sub>	подстрочное написание символов	inline
<sup></sup>	надстрочное написание символов	inline
<time></time>	дата / время документа или статьи	inline
<abbr></abbr>	аббревиатура или акроним	none
<address></address>	контактные данные автора документа или статьи	block
<bdi></bdi>	изолирует текст, читаемый справа налево	inline
<bdo></bdo>	задаёт направление написания текста	inline

<ruby></ruby>	контейнер для Восточно-Азиатских символов и их расшифровки	inline
<rp></rp>	тег для скобок вокруг символов	none
<rt></rt>	расшифровка символов	block

## Ссылки

Ссылки являются основой гипертекстовых документов и позволяют переходить с одной веб-страницы на другую. Особенность их состоит в том, что сама ссылка может вести не только на HTML-файлы, но и на файл любого типа, причем этот файл может размещаться совсем на другом сайте. Главное, чтобы к документу, на который делается ссылка, был доступ.

Ссылки можно поделить на две категории:

1) **ссылки на внешние ресурсы** — создаются с помощью тега **<link>** и используются для расширения возможностей текущего документа при обработке браузером;

2) **гиперссылки** — ссылки на другие ресурсы, которые пользователь может посетить или загрузить.

Для создания ссылки необходимо сообщить браузеру, что является ссылкой, а также указать адрес документа, на который следует сделать ссылку. Оба действия выполняются с помощью тега **<a>**, который имеет единственный обязательный атрибут **href**. В качестве значения используется адрес документа (URL). Общий синтаксис создания ссылок следующий:

```
<a href="http://site.ru">указатель ссылки</a>
```

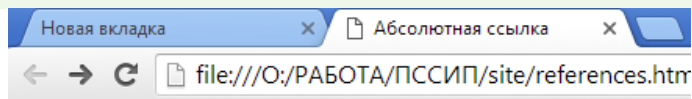
*Ссылка состоит из двух частей — указателя и адресной части. Указатель ссылки представляет собой фрагмент текста или изображение, видимые для пользователя. Адресная часть ссылки пользователю не видна, она представляет собой адрес ресурса, к которому необходимо перейти.*

Адрес ссылки может быть абсолютным и относительным.

*Абсолютные адреса работают везде и всюду независимо от имени сайта или веб-страницы, где прописана ссылка. Начинаются они с указания протокола передачи данных (так, для веб-страниц это обычно HTTP (HyperText Transfer Protocol, протокол передачи гипертекста), соответственно, абсолютные ссылки начинаются с ключевого слова http://) и должны содержать имя сайта.*

Пример использования абсолютных ссылок:

```
<!DOCTYPE HTML>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Абсолютная ссылка</title>
</head>
<body>
  <p><a href="http://www.mrk-bsuir.by/">Минский радиотехнический колледж</a></p>
</body>
</html>
```



[Минский радиотехнический колледж](#)

Абсолютный путь указывает точное местоположение файла в пределах всей структуры папок на компьютере (сервере). Абсолютный путь к файлу даёт доступ к файлу со сторонних ресурсов и содержит следующие компоненты:

- 1) протокол, например, http (опционально);
- 2) домен (доменное имя или IP-адрес компьютера);
- 3) папка (имя папки, указывающей путь к файлу);
- 4) файл (имя файла).

*Существует два вида записи абсолютного пути — с указанием протокола и без него:*

```
http://site.ru/pages/tips/tips1.html  
//site.ru/pages/tips/tips1.html
```

*При отсутствии имени файла будет загружаться веб-страница, которая задана по умолчанию в настройках веб-сервера (так называемый индексный файл).*

**Относительные** ссылки, как следует из их названия, построены относительно текущего документа или адреса и, соответственно, ведут отсчет от корня сайта или текущего документа.

*Относительные ссылки используются при создании ссылок на другие документы на одном и том же сайте. Когда браузер не находит в ссылке протокол http://, он выполняет поиск указанного документа на том же сервере.*

Относительный путь содержит следующие компоненты:

- 1) папка (имя папки, указывающей путь к файлу);
- 2) файл (имя файла).

Путь для относительных ссылок имеет три специальных обозначения:

/ указывает на корневую директорию и говорит о том, что нужно начать путь от корневого каталога документов и идти вниз до следующей папки;

./ указывает на текущую папку;

../ подняться на одну папку (директорию) выше

*Главное отличие относительного пути от абсолютного в том, что относительный путь не содержит имени корневой папки и родительских папок, что делает адрес короче, и в случае переезда с одного домена на другой не нужно прописывать новый абсолютный адрес.*

Примеры относительных адресов:

/, /demo/ - эти две ссылки называются неполные и указывают веб-серверу загружать файл index.html (или default.html), который находится в корне сайта или папке demo. Если файл index.html отсутствует, браузер, как правило, показывает список файлов, находящихся в данном каталоге;

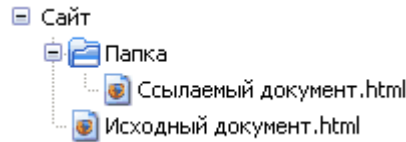
Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.

Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

**/images/pic.gif** - слэш перед адресом говорит о том, что адресация начинается от корня сайта. Ссылка ведет на рисунок pic.gif, который находится в папке images. А та в свою очередь размещена в корне сайта;

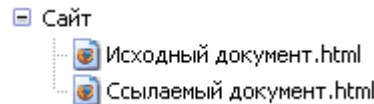
**../help/me.html** - две точки перед именем указывают браузеру перейти на уровень выше в списке каталогов сайта;

**manual/info.html** - если перед именем папки нет никаких дополнительных символов, вроде точек, то она размещена внутри текущего каталога, т.е. отсчет ведется от текущей папки. Иногда можно встретить в адресе ссылки путь в виде **./file/doc.html**;



```
<a href="Папка/Ссылаемый документ.html">Ссылка</a>
```

**doc.html** - файлы располагаются в одной папке.



Пример относительной ссылки:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>Ссылки</title>
</head>
<body>
<p><a href="./site/1.html">Научная статья</a></p>
<p><a href="./site/2.html">Как создать сайт?</a></p>
</body>
</html>
```

На сайте в именах файлов не следует использовать русские символы с пробелами, да еще и в разном регистре.

Любая ссылка на веб-странице может находиться в одном из следующих состояний:

**1) непосещенная ссылка.** Такое состояние характеризуется для ссылок, которые еще не открывали. По умолчанию непосещенные текстовые ссылки изображаются синего цвета и с подчеркиванием;

**2) активная ссылка.** Ссылка помечается как активная в момент ее открытия. Поскольку время между нажатием на ссылку и началом загрузки нового документа достаточно мало, подобное состояние ссылки весьма кратковременно. Активной ссылка становится также, при ее выделении с помощью клавиатуры. Цвет такой ссылки по умолчанию красный;

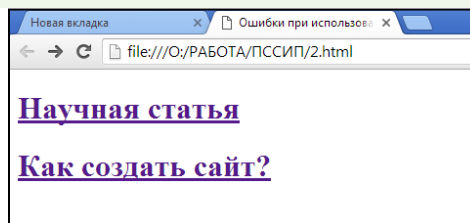
**3) посещенная ссылка.** Как только пользователь открывает документ, на который указывает ссылка, она помечается как посещенная и меняет свой цвет на фиолетовый, установленный по умолчанию.

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Любая ссылка является встроенным элементом, поэтому для нее действуют те же правила, что и для встроенных элементов. А именно, **нельзя размещать внутри тега <a> блочные элементы, но допустимо делать наоборот, и вкладывать ссылку в блочный контейнер.**

Пример:

```
1      <!DOCTYPE HTML>
2      <html>
3      <head>
4      <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
5      <title>Ошибки при использовании ссылок</title>
6      </head>
7      <body>
8      <p><a href="./site/1.html"><h1>Научная статья</h1></a></p>
9      <p><h1><a href="./site/2.html">Как создать сайт?</a></h1></p>
10     </body>
11     </html>
```



В строке 8 данного примера содержится типичная ошибка — тег `<h1>` располагается внутри контейнера `<a>`. Поскольку `<h1>` это блочный элемент, то его недопустимо вкладывать внутрь ссылки. В строке 9 этого же примера показан корректный вариант.

## Якоря

Якорем называется закладка с уникальным именем на определенном месте веб-страницы, предназначенная для создания перехода к ней по ссылке. Якоря удобно применять в документах большого объема, чтобы можно было быстро переходить к нужному разделу.

Внутренние ссылки также создаются при помощи тега `<a>` с разницей в том, что атрибут **href** содержит имя указателя — так называемый якорь, а не URI-адрес. Перед именем указателя всегда ставится знак `#`.

Следующая разметка создаст оглавление с быстрыми переходами на соответствующие разделы:

```
<h1>Времена года</h1>
<h2>Оглавление</h2>
<a href="#p1">Лето</a> <!--создаём якорь, указав #id элемента-->
<a href="#p2">Осень</a>
<a href="#p3">Зима</a>
<a href="#p4">Весна</a>
<p id="p1">...</p> <!--добавляем соответствующий id элементу-->
<p id="p2">...</p>
<p id="p3">...</p>
<p id="p4">...</p>
```

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Ссылку можно также сделать на закладку, находящуюся в другой веб-странице и даже другом сайте. Для этого в атрибуте **href** тега **<a>** надо указать адрес документа и в конце добавить символ решетки **#** и имя закладки.

Пример:

```
<p><a href="text.html#bottom">Перейти к нижней части текста</a></p>
```

В данном примере показано создание ссылки на файл *text.html*, при открытии этого файла происходит переход на закладку с именем *bottom*.

По клику по ссылке можно не только переходить на другие страницы и скачивать файлы, но и совершать звонки на телефоны, отправлять сообщения или звонить по скайпу.

ссылка на телефонный номер

```
<a href="tel:+74951234567">+7 (495) 123-45-67</a>
```

ссылка на адрес электронной почты

```
<a href="mailto:example@mail.ru">example@mail.ru</a>
```

ссылка на скайп (позвонить)

```
<a href="skype:имя-пользователя?call">Skype</a>
```

ссылка на скайп (открыть чат)

```
<a href="skype:имя-пользователя?chat">Skype</a>
```

ссылка на скайп (добавить в список контактов)

```
<a href="skype:имя-пользователя?add">Skype</a>
```

ссылка на скайп (отправить файл)

```
<a href="skype:имя-пользователя?sendfile">Skype</a>
```

Можно также автоматически добавить тему сообщения, присоединив к адресу электронной почты через символ вопроса (?) параметр *subject=тема сообщения*. При запуске почтовой программы поле Тема (*Subject*) будет заполнено автоматически.

```
<p><a href="mailto: office@mrk-bsuir.by?subject=Вопрос по HTML">Задавайте вопросы по электронной почте</a></p>
```

Элемент **<a>** поддерживает глобальные атрибуты и собственные.

Атрибут	Значения	Описание
download	"пустой" название файла	Указывает браузеру, что материал по ссылке необходимо скачать. Если указано название файла, в диалоговом окне загрузки будет предложено сохранить файл именно под этим названием вместо названия файла с сайта.
href	URI адрес	Определяет адрес документа, на который ведет ссылка.
hreflang	language_code	Указывает на язык документа по ссылке.
media	media_query	Указывает на устройство, для которого оптимизирован документ по ссылке.



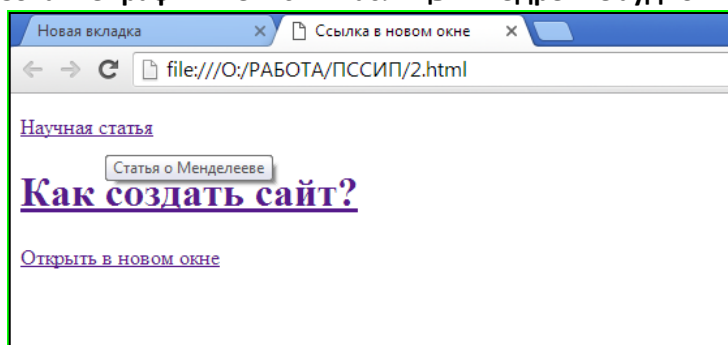
**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

rel	alternate author bookmark external help license next nofollow noreferrer noopener prev search tag	Определяет отношение между текущим документом и документом по ссылке. Указывает чем является документ по ссылке для текущего.
target	_blank _parent _self _top <i>имя фрейма</i>	Определяет где будет открыт ссылаемый документ: _blank - в новой вкладке или окне; _parent - в родительском окне; _self - в текущем окне (используется по умолчанию); _top - на весь экран.
type	<i>media_type</i>	Указывает на тип ссылаемого документа.
<b>Устаревшие атрибуты</b>		
charset	<i>char_encoding</i>	Указывает на кодировку документа по ссылке.
coords	<i>coordinates</i>	Определяет координаты ссылки. Используется вместе с атрибутом shape, например, чтобы задать размер, форму и положение ссылки на изображении <img>.
name	<i>название элемента</i>	Задаёт название элемента для якорной ссылки. В HTML5 для этой цели используется универсальный атрибут id.
rev	<i>текстовое значение</i>	Определяет отношение между текущим документом и документом по ссылке. Противоположность rel. Указывает чем является текущий документ для ссылаемого.
shape	default rect circle poly	Определяет форму ссылки. Используется вместе с атрибутом coords, например, чтобы задать размер, форму и положение ссылки на изображении <img>.

**Пример:**

```
<p><a href="./site/1.html" title="Статья о Менделееве">Научная статья</a></p>
```

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты



Средствами тегов HTML убрать подчеркивание у ссылок не представляется возможным. Поэтому для этой цели используются каскадные таблицы стилей (Cascading Style Sheets, CSS).

## Использование графики

Возможность использования графики в HTML трудно переоценить. Правильно подобранная и размещенная на Web-странице графика делает её визуально привлекательной и, что самое главное, передаёт одну из основных идей документа.

Изображения на web-страницах могут использоваться двумя способами: в качестве фона и в качестве самостоятельного изображения. Рекомендуется использование трех форматов графики: JPEG, GIF и PNG. Именно их поддерживают все браузеры, для остальных форматов могут потребоваться специальные средства.

Для добавления к web-странице файла с изображением используется тег **<img>**, с обязательным атрибутом **src**, который будет указывать на файл с изображением.

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

Пример вставки изображения



### Необязательные атрибуты тега <IMG>:

**ALT** – позволяет указать текст, который будет выводиться вместо изображения браузерами, неспособными представлять графику. Атрибут alt используется как альтернативный текст в случае отсутствия изображения, а не как текст при наведении мышки. Для показа текста при наведении мышки на изображение используйте атрибут title;

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

Атрибут	Значение	Описание
---------	----------	----------

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

Align	left right top middle bottom	Определяет положение изображения относительно окружающего его текста. Устарел. Используйте CSS
Border	<i>пиксели</i>	Численное значение аргумента определяет толщину рамки вокруг изображения. Устарел. Используйте CSS
Height	<i>пиксели %</i>	Определяет высоту изображения
Hspace	<i>пиксели</i>	Численное значение этого атрибута задает горизонтальное расстояние между вертикальной границей страницы и изображением, а также между изображением и огибающим его текстом. Устарел. Используйте CSS
Ismap	ismap	Определяет изображение как карту изображение на стороне сервера. Очень редко используется.
Longdesc	<i>URL</i>	Определяет путь к документу, который содержит длинное описание изображения
Usemap	<i>#название_карты</i>	Определяет изображение как карту изображение на стороне сервера.
Vspace	<i>пиксели</i>	Численное значение этого атрибута задает вертикальное расстояние между строками текста и изображением. Устарел. Используйте CSS
Width	<i>пиксели %</i>	Определяет ширину изображения

```
<A href="./site/picture.jpg"></A>
```

Ширину и высоту изображения можно менять как в меньшую, так и большую сторону.

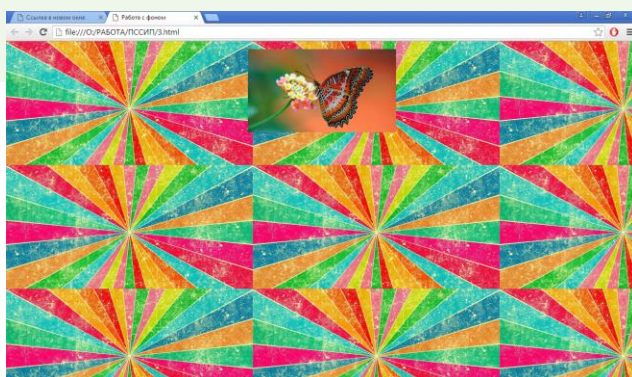
Для использования изображения в качестве фона используется атрибут **background** с адресом фонового рисунка. Особенностью тега background является то что он при открытии изображения помещает его в полный размер в левый верхний угол экрана и начинает его тиражировать на весь экран, что позволяет добиваться довольно интересных эффектов.

Пример:

```
<!DOCTYPE HTML>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>Работа с фоном</title>
</head>

  <body background="site/tsvet-fona-html.png">
<p><center> </center></p>

</body>
</html>
```



Для построения горизонтальной линии используется тег `<hr>`.

```
<hr align="center" color="#666666" width="400">
```

Ссылки являются основой гипертекстовых документов и позволяют переходить с одной веб-страницы на другую. Особенность их состоит в том, что сама ссылка может вести не только на HTML-файлы, но и на файл любого типа, причем этот файл может размещаться совсем на другом сайте. Главное, чтобы к документу, на который делается ссылка, был доступ.

Якорем называется закладка с уникальным именем на определенном месте веб-страницы, предназначенная для создания перехода к ней по ссылке. Якоря удобно применять в документах большого объема, чтобы можно было быстро переходить к нужному разделу.

Рисунки можно использовать в качестве ссылок. Для этого вместо названия ссылки нужно вставить графический элемент, а чтобы при наведении на изображение всплывала подсказка куда приведет эта ссылка, необходимо вставить необязательный атрибут `alt` и в значении указывать название информационного ресурса. Атрибут `href` тега `<a>` задает путь к документу, на который указывает ссылка, а `src` тега `<img>` — путь к графическому файлу.

Вокруг изображения-ссылки автоматически добавляется рамка толщиной один пиксел и цветом, совпадающим с цветом текстовых ссылок. Чтобы убрать рамку, следует у тега `<img>` установить атрибут `border="0"`

*Пример:*

```
<A href="./site/picture.jpg"></A>
```



## Списки

Списком называется взаимосвязанный набор отдельных фраз или предложений, которые начинаются с маркера или цифры. Списки предоставляют возможность упорядочить и систематизировать разные данные и представить их в наглядном и удобном для пользователя виде.

Современным стандартом HTML предусмотрено три основных вида списков:

- 1) маркированный список;
- 2) нумерованный список;
- 3) список определений.

Каждый список представляет собой контейнер, внутри которого располагаются элементы списка или пары термин-определение. Элементы списка ведут себя как блочные элементы, располагаясь друг под другом и занимая всю ширину блока-контейнера. *Каждый*

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**  
*элемент списка имеет дополнительный блок, расположенный сбоку, который не участвует в компоновке.*

**Маркированный** (неупорядоченный) список определяется тем, что перед каждым элементом списка добавляется небольшой маркер, обычно в виде закрашенного кружка. Сам список формируется с помощью контейнера `<ul>`, а каждый пункт списка начинается с тега `<li>` (item list), как показано ниже.

```
<ul>
  <li>Microsoft</li>
  <li>Google</li>
  <li>Apple</li>
  <li>IBM</li>
</ul>
```

В списке непременно должен присутствовать закрывающий тег `</ul>`, иначе возникнет ошибка. Закрывающий тег `</li>` и не обязателен, но следует его добавлять, чтобы четко разделять элементы списка.

Отступы сверху, снизу и слева от списка добавляются автоматически. Маркеры могут принимать один из трех видов:

- круг (по умолчанию);
- окружность;
- квадрат.

Для выбора стиля маркера используется атрибут `type` тега `<ul>`. Допустимые значения приведены в таблице.

Тип списка	Код HTML	Пример
Список с маркерами в виде круга	<pre>&lt;ul type="disc"&gt; &lt;li&gt;...&lt;/li&gt; &lt;/ul&gt;</pre>	<ul style="list-style-type: none"><li>• Первый</li><li>• Второй</li><li>• Третий</li></ul>
Список с маркерами в виде окружности	<pre>&lt;ul type="circle"&gt; &lt;li&gt;...&lt;/li&gt; &lt;/ul&gt;</pre>	<ul style="list-style-type: none"><li>○ Первый</li><li>○ Второй</li><li>○ Третий</li></ul>
Список с квадратными маркерами	<pre>&lt;ul type="square"&gt; &lt;li&gt;...&lt;/li&gt; &lt;/ul&gt;</pre>	<ul style="list-style-type: none"><li>▪ Первый</li><li>▪ Второй</li><li>▪ Третий</li></ul>

*Вид маркеров может незначительно различаться в разных браузерах, а также при смене шрифта и размера текста.*

**Нумерованные** (упорядоченные) списки представляют собой набор элементов с их порядковыми номерами. Вид и тип нумерации зависит от атрибутов тега `<ol>`, который и применяется для создания списка. Каждый пункт нумерованного списка обозначается тегом `<li>`.

По умолчанию применяется список с арабскими числами (1, 2, 3,...). Также добавляются автоматические отступы сверху, снизу и слева от текста.

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

В качестве нумерующих элементов могут выступать следующие значения:

- арабские числа (1, 2, 3, ...);
- прописные латинские буквы (A, B, C, ...);
- строчные латинские буквы (a, b, c, ...);
- прописные римские числа (I, II, III, ...);
- строчные римские числа (i, ii, iii, ...).

Для указания типа нумерованного списка применяется атрибут type тега <ol>. Его возможные значения приведены в таблице

Тип списка	Код HTML	Пример
Арабские числа	<code>&lt;ol type="1"&gt;</code> <code>&lt;li&gt;...&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	1. Чебурашка 2. Крокодил Гена 3. Шапокляк
Прописные буквы латинского алфавита	<code>&lt;ol type="A"&gt;</code> <code>&lt;li&gt;...&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	A. Чебурашка B. Крокодил Гена C. Шапокляк
Строчные буквы латинского алфавита	<code>&lt;ol type="a"&gt;</code> <code>&lt;li&gt;...&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	a. Чебурашка b. Крокодил Гена c. Шапокляк
Римские числа в верхнем регистре	<code>&lt;ol type="I"&gt;</code> <code>&lt;li&gt;...&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	I. Чебурашка II. Крокодил Гена III. Шапокляк
Римские числа в нижнем регистре	<code>&lt;ol type="i"&gt;</code> <code>&lt;li&gt;...&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	i. Чебурашка ii. Крокодил Гена iii. Шапокляк

Чтобы начать список с определенного значения, используется атрибут **start** тега <ol>. При этом не имеет значения, какой тип списка установлен с помощью type, атрибут start одинаково работает и с римскими и с арабскими числами.

Список определений состоит из двух элементов — термина и его определения. Сам список задается с помощью контейнера **<dl>** (definition list), термин — тегом **<dt>** (definition term), а его определение — с помощью тега **<dd>** (definition description).

*Список определений хорошо подходит для расшифровки терминов, создания глоссария, словаря, справочника и т.д.*

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>Списки</title>
</head>
<body>
<hr>
<ul type="square">
```



Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.  
Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

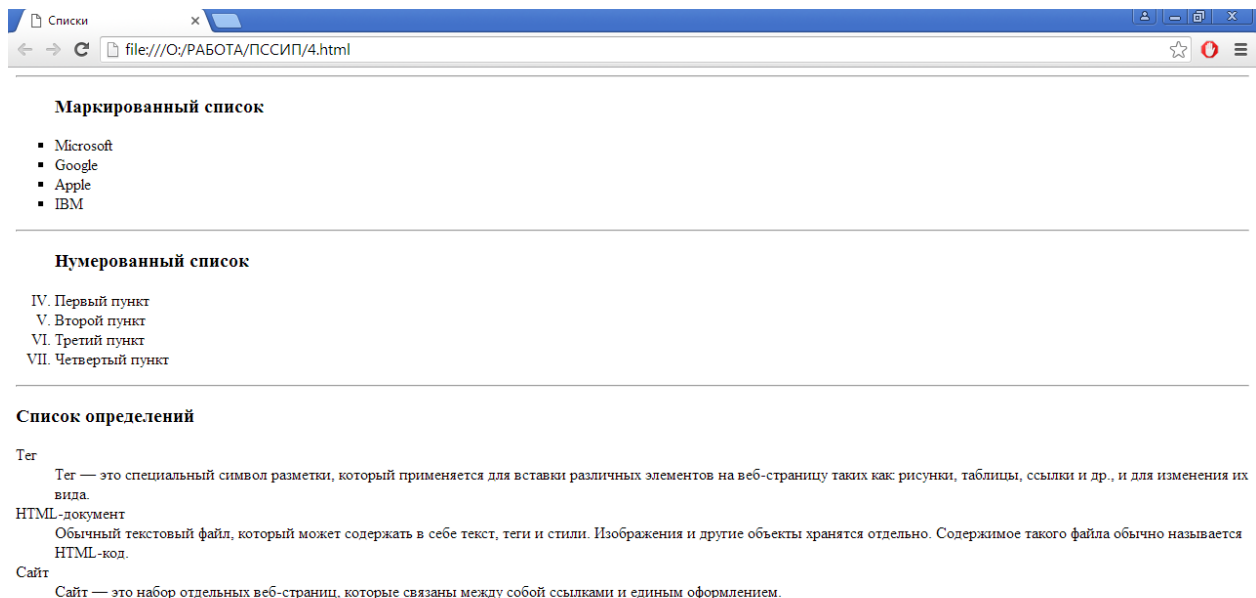
```
<h3>Маркированный список</h3>
<li>Microsoft</li>
  <li>Google</li>
  <li>Apple</li>
  <li>IBM</li>
</ul>
<hr>

<ol type="I" start="4">
<h3>Нумерованный список</h3>
  <li>Первый пункт</li>
  <li>Второй пункт</li>
  <li>Третий пункт</li>
  <li>Четвертый пункт</li>
</ol>
<hr>

<h3>Список определений</h3>

<dl>
  <dt>Ter</dt>
  <dd>Ter — это специальный символ разметки, который применяется для
    вставки различных элементов на веб-страницу таких как: рисунки,
    таблицы, ссылки и др., и для изменения их вида.</dd>
  <dt>HTML-документ</dt>
  <dd>Обычный текстовый файл, который может содержать в себе текст,
    теги и стили. Изображения и другие объекты хранятся отдельно.
    Содержимое такого файла обычно называется HTML-код.</dd>
  <dt>Сайт</dt>
  <dd>Сайт — это набор отдельных веб-страниц, которые связаны между собой
    ссылками и единым оформлением.</dd>
</dl>

</body>
</html>
```



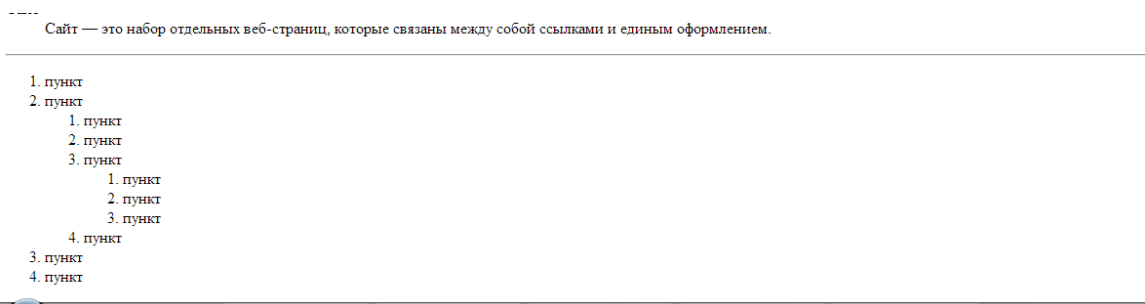
Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

**Вложенный список** – используется, когда возможностей простых списков не хватает, например, при создании оглавления никак не обойтись без вложенных пунктов. Разметка для вложенного списка будет следующей:

```
<ul>
<li>Пункт 1.</li>
  <li>Пункт 2.
    <ul>
      <li>Подпункт 2.1.</li>
      <li>Подпункт 2.2.
        <ul>
          <li>Подпункт 2.2.1.</li>
          <li>Подпункт 2.2.2.</li>
        </ul>
      </li>
      <li>Подпункт 2.3.</li>
    </ul>
  </li>
<li>Пункт 3.</li>
</ul>
```

**Многоуровневый список** html страницы может использовать как маркированные, так и нумерованные списки, а также их сочетание. Многоуровневый список получается путем вложения одного списка в тело другого. Многоуровневый список используется для отображения элементов списка на разных уровнях с различными отступами. Разметка для многоуровневого нумерованного списка будет следующей:

```
<ol>
  <li>пункт</li> <!--1.-->
  <li>пункт
    <ol>
      <li>пункт</li> <!--2.1.-->
      <li>пункт</li> <!--2.2.-->
      <li>пункт
        <ol>
          <li>пункт</li> <!--2.3.1.-->
          <li>пункт</li> <!--2.3.2.-->
          <li>пункт</li> <!--2.3.3.-->
        </ol>
      </li> <!--2.3.-->
      <li>пункт</li> <!--2.4.-->
    </ol>
  </li> <!--2.-->
  <li>пункт</li> <!--3.-->
  <li>пункт</li> <!--4.-->
</ol>
```



Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.

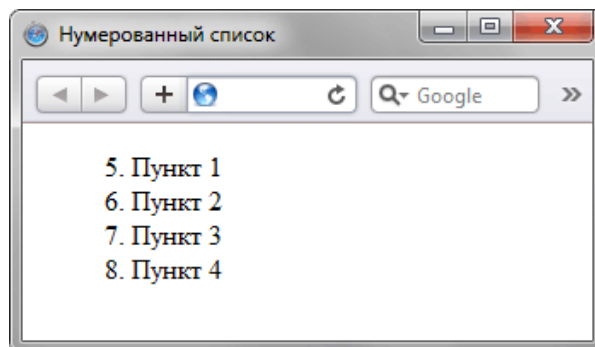
Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Такая разметка по умолчанию создаст для каждого вложенного списка новую нумерацию, начинающуюся с единицы. Чтобы сделать вложенную нумерацию, нужно использовать следующие свойства:

- **counter-reset** сбрасывает один или несколько счётчиков, задавая значение для сброса;
- **counter-increment** задаёт значение приращения счётчика, т.е. с каким шагом будет нумероваться каждый последующий пункт;
- **content** — генерируемое содержимое, в данном случае отвечает за вывод номера перед каждым пунктом списка.

Пример:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Нумерованный список</title>
  <style>
    li { list-style-type: none; } /* Убираем исходную нумерацию у списка */
    ol { counter-reset: list 4; } /* Инициализируем счетчик */
    ol li:before {
      counter-increment: list; /* Увеличиваем значение счетчика */
      content: counter(list) ". "; /* Выводим число */
    }
  </style>
</head>
<body>
  <ol>
    <li>Пункт 1</li>
    <li>Пункт 2</li>
    <li>Пункт 3</li>
    <li>Пункт 4</li>
  </ol>
</body>
</html>
```



## Таблицы

Благодаря универсальности таблиц, большому числу параметров, управляющих их видом, таблицы надолго стали определенным стандартом для верстки веб-страниц. Таблицы применяются для размещения табличных данных, а слои — для верстки и оформления.

Создание таблицы

Для добавления таблицы на веб-страницу используется тег **<table>**. Этот элемент служит контейнером для элементов, определяющих содержимое таблицы. Любая таблица состоит из строк и ячеек, которые задаются соответственно с помощью тегов **<tr>** и **<td>**. Таблица должна содержать хотя бы одну ячейку.

Допускается вместо тега **<td>** использовать тег **<th>**. Текст в ячейке, оформленной с помощью тега **<th>**, отображается браузером шрифтом жирного начертания и выравнивается по центру ячейки. В остальном, разницы между ячейками, созданными через теги **<td>** и **<th>** нет.

Тот факт, что таблицы применяются достаточно часто и не только для отображения табличных данных обязан не только их гибкости и универсальности, но и обилию атрибутов тегов **<table>**, **<tr>** и **<td>**.

#### Атрибуты тега <table>:

**align** - задает выравнивание таблицы по краю окна браузера. Допустимые значения: **left** — выравнивание таблицы по левому краю, **center** — по центру и **right** — по правому краю. Когда используются значения **left** и **right**, текст начинает обтекать таблицу сбоку и снизу. *Результат будет замечен только в том случае, если ширина таблицы не занимает всю доступную область, другими словами, меньше, чем 100%. На самом деле align не только устанавливает выравнивание, но и заставляет текст обтекать таблицу с других сторон аналогично поведению тега <img>;*

**bgcolor** - устанавливает цвет фона таблицы;

**border** - устанавливает толщину границы в пикселах вокруг таблицы. При наличии этого атрибута также отображаются границы между ячейками;

**cellpadding** - определяет расстояние между границей ячейки и ее содержимым. Этот атрибут добавляет пустое пространство к ячейке, увеличивая тем самым ее размеры. Без cellpadding текст в таблице «налипает» на рамку, ухудшая тем самым его восприятие. *Добавление же cellpadding позволяет улучшить читабельность текста. При отсутствии границ особого значения этот атрибут не имеет, но может помочь, когда требуется установить пустой промежуток между ячейками;*

**cellspacing** - задает расстояние между внешними границами ячеек. Если установлен атрибут border, толщина границы принимается в расчет и входит в общее значение;

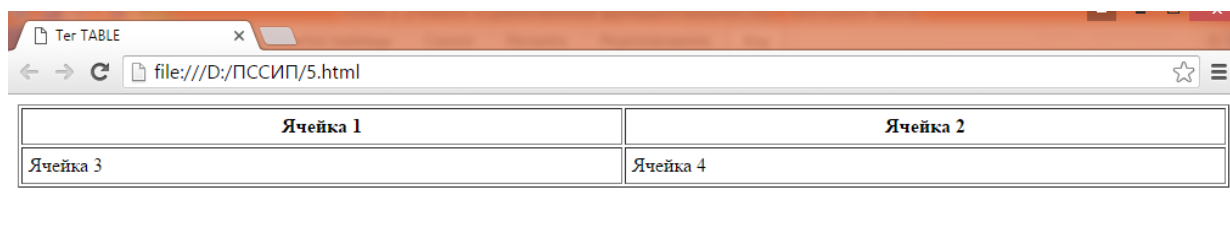
**cols** - атрибут cols указывает количество столбцов в таблице, помогая браузеру в подготовке к ее отображению. Без этого атрибута таблица будет показана только после того, как все ее содержимое будет загружено в браузер и проанализировано. Использование атрибута cols позволяет несколько ускорить отображение содержимого таблицы;

**rules** - сообщает браузеру, где отображать границы между ячейками. По умолчанию рамка рисуется вокруг каждой ячейки, образуя тем самым сетку. В дополнение можно указать отображать линии между колонками (значение cols), строками (rows) или группами (groups), которые определяются наличием тегов **<thead>**, **<tfoot>**, **<tbody>**, **<colgroup>** или **<col>**. Толщина границы указывается с помощью атрибута border.

**width** - задает ширину таблицы. *Если общая ширина содержимого превышает указанную ширину таблицы, то браузер будет пытаться «втиснуться» в заданные размеры за счет форматирования текста. В случае, когда это невозможно, например, в таблице находятся изображения, атрибут width будет проигнорирован, и новая ширина таблицы будет вычислена на основе ее содержимого.*

#### Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset= windows-1251">
<title>Ter TABLE</title>
</head>
<body>
<table border="1" width="100%" cellpadding="5">
<tr>
<th>Ячейка 1</th>
<th>Ячейка 2</th>
</tr>
<tr>
<td>Ячейка 3</td>
<td>Ячейка 4</td>
</tr>
</table>
</body>
</html>
```



Ячейка 1	Ячейка 2
Ячейка 3	Ячейка 4

### Атрибуты тега <td>

Каждая ячейка таблицы, задаваемая через тег <td>, в свою очередь тоже имеет свои атрибуты, часть из которых совпадает с атрибутами тега <table>:

#### **align, bgcolor;**

**colspan** - устанавливает число ячеек, которые должны быть объединены по горизонтали. Этот атрибут имеет смысл для таблиц, состоящих из нескольких столбцов. Например, как для таблицы, показанной на рисунке:

ячейка 1	
ячейка 2	ячейка 3

*В приведенной таблице содержатся две строки и две колонки, причем верхние горизонтальные ячейки объединены с помощью атрибута colspan.*

**height** - браузер сам устанавливает высоту таблицы и ее ячеек исходя из их содержимого. Однако при использовании атрибута height высота ячеек будет изменена. Здесь возможны два варианта. Если значение height меньше, чем содержимое ячейки, то этот атрибут будет проигнорирован. В случае, когда установлена высота ячейки, превышающая ее содержимое, добавляется пустое пространство по вертикали;

**rowspan** - устанавливает число ячеек, которые должны быть объединены по вертикали. Этот атрибут имеет смысл для таблиц, состоящих из нескольких строк. Например, как для таблицы, показанной на рисунке (*вертикальное объединение ячеек*):

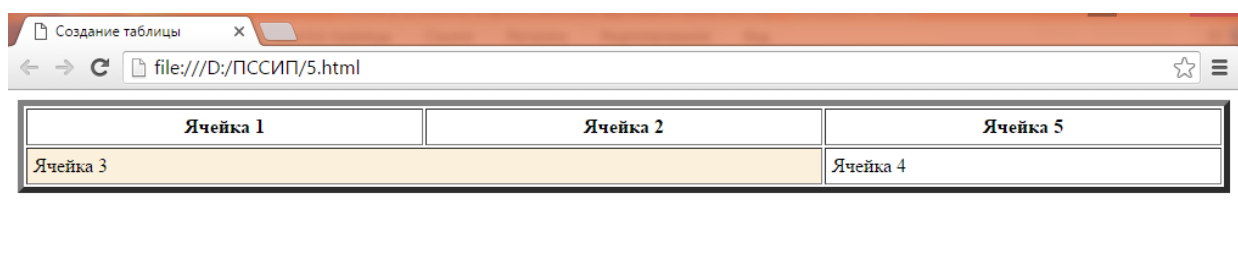
ячейка 1	ячейка 2
	ячейка 3

**valign** - устанавливает вертикальное выравнивание содержимого ячейки. По умолчанию содержимое ячейки располагается по ее вертикали в центре. Возможные значения: top — выравнивание по верхнему краю строки, middle — выравнивание по середине, bottom — выравнивание по нижнему краю, baseline — выравнивание по базовой линии, при этом происходит привязка содержимого ячейки к одной линии;

**width** - задает ширину ячейки. Суммарное значение ширины всех ячеек может превышать общую ширину таблицы только в том случае, если содержимое ячейки превышает указанную ширину.

Пример:

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>Создание таблицы</title>
</head>
<body>
  <table border="5" width="100%" cellpadding="5">
    <tr>
      <th>Ячейка 1</th>
      <th>Ячейка 2</th>
      <th>Ячейка 5</th>
    </tr>
    <tr>
      <td colspan="2" bgcolor="#FBF0DB">Ячейка 3</td>
      <td>Ячейка 4</td>
    </tr>
  </table>
</body>
</html>
```



Особенности таблиц:

У каждого параметра таблицы есть свое значение установленное по умолчанию. Это означает, что если какой-то атрибут пропущен, то неявно он все равно присутствует, причем с некоторым значением. Из-за чего вид таблицы может оказаться совсем другим, нежели предполагал разработчик. Чтобы понимать, что можно ожидать от таблиц, следует знать их явные и неявные особенности, которые перечислены далее.

1. Одну таблицу допускается помещать внутри ячейки другой таблицы. Это требуется для представления сложных данных или в том случае, когда одна таблица выступает в роли модульной сетки, а вторая, внутри нее, уже как обычная таблица.

2. Размеры таблицы изначально не установлены и вычисляются на основе содержимого ячеек. Например, общая ширина определяется автоматически исходя из



**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.**

**Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**  
суммарной ширины содержимого ячеек плюс ширина границ между ячейками, поля вокруг содержимого, устанавливаемые через атрибут `cellpadding` и расстояние между ячейками, которые определяются значением `cellspacing`.

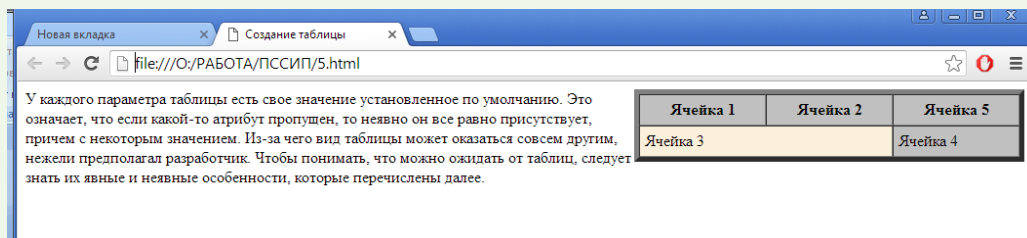
3. Если для таблицы задана ее ширина в процентах или пикселах, то содержимое таблицы подстраивается под указанные размеры. Так, браузер автоматически добавляет переносы строк в текст, чтобы он полностью поместился в ячейку, и при этом ширина таблицы осталась без изменений. Бывает, что ширину содержимого ячейки невозможно изменить, как это, например, происходит с рисунками. В этом случае ширина таблицы увеличивается, несмотря на указанные размеры.

4. Пока таблица не загрузится полностью, ее содержимое не начнет отображаться. Дело в том, что браузер, прежде чем показать содержимое таблицы, должен вычислить необходимые размеры ячеек, их ширину и высоту. А для этого необходимо знать, что в этих ячейках находится. Поэтому браузер и ожидает, пока загрузится все, что находится в ячейках, и только потом отображает таблицу.

### Выравнивание таблиц (и обтекание текстом)

Пример:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>Создание таблицы</title>
</head>
<body>
<table align="right" border="5" bgcolor="#c0c0c0" cellspacing="0" width="400"
cellpadding="5">
<tr>
<th>Ячейка 1</th>
<th>Ячейка 2</th>
<th>Ячейка 5</th>
</tr>
<tr>
<td colspan="2" bgcolor="#FBF0DB">Ячейка 3</td>
<td>Ячейка 4</td>
</tr>
</table>
<p>У каждого параметра таблицы есть свое значение установленное по умолчанию. Это
означает, что если какой-то атрибут пропущен, то неявно он все равно присутствует,
причем с некоторым значением. Из-за чего вид таблицы может оказаться совсем другим,
нежели предполагал разработчик. Чтобы понимать, что можно ожидать от таблиц, следует
знать их явные и неявные особенности, которые перечислены далее.</p>
</body>
</html>
```



По умолчанию таблица формируется в виде сетки, при этом в каждой строке таблицы содержится одинаковое количество ячеек. Такой вариант вполне подходит для формирования простых таблиц, но совершенно не годится для тех случаев, когда

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**  
*предстоит сделать сложную таблицу. В подобных ситуациях применяются два основных метода: объединение ячеек и вложенные таблицы.*

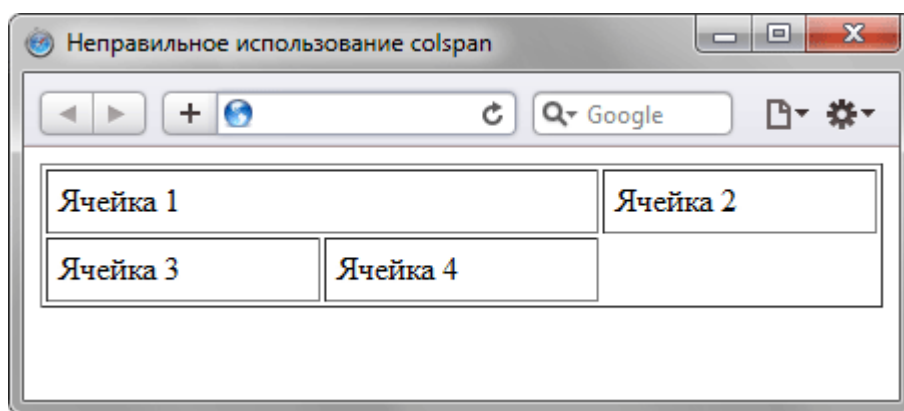
### Объединение ячеек

Для объединения двух и более ячеек в одну используются атрибуты `colspan` и `rowspan` тега `<td>`. Атрибут `colspan` устанавливает число ячеек объединяемых по горизонтали. Аналогично работает и атрибут `rowspan`, с тем лишь отличием, что объединяет ячейки по вертикали. Перед добавлением атрибутов проверьте число ячеек в каждой строке, чтобы не возникло ошибок. Так, `<td colspan="3">` заменяет три ячейки, поэтому в следующей строке должно быть три тега `<td>` или конструкция вида `<td colspan="2">...</td><td>...</td>`. Если число ячеек в каждой строке не будет совпадать, появятся пустые фантомные ячейки.

В примере приведен хотя и валидный, но неверный код, в котором как раз проявляется подобная ошибка.

### Пример (неправильное использование colspan):

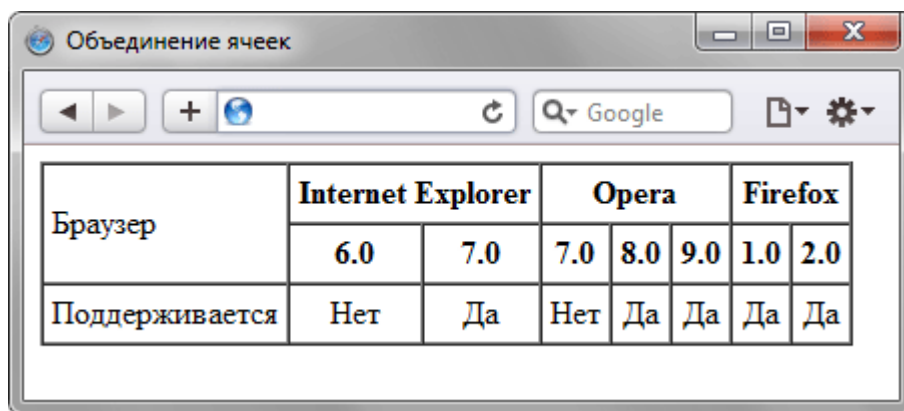
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Неправильное использование colspan</title>
</head>
<body>
<table border="1" cellpadding="5" width="100%">
<tr>
<td colspan="2">Ячейка 1</td>
<td>Ячейка 2</td>
</tr>
<tr>
<td>Ячейка 3</td>
<td>Ячейка 4</td>
</tr>
</table>
</body>
</html>
```



*В первой строке примера задано три ячейки, две из них объединены с помощью атрибута `colspan`, а во второй строке добавлено только две ячейки. Из-за этого возникает дополнительная ячейка, которая отображается в браузере. Ее хорошо видно на рисунке.*

Пример (правильное использование атрибутов `colspan` и `rowspan`):

```
!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Объединение ячеек</title>
</head>
<body>
<table border="1" cellpadding="4" cellspacing="0">
<tr>
<td rowspan="2">Браузер</td>
<th colspan="2">Internet Explorer</th>
<th colspan="3">Opera</th>
<th colspan="2">Firefox</th>
</tr>
<tr>
<th>6.0</th><th>7.0</th><th>7.0</th><th>8.0</th><th>9.0</th><th>1.0</th><th>2.0</th>
</tr>
<tr align="center">
<td>Поддерживается</td>
<td>Нет</td><td>Да</td><td>Нет</td><td>Да</td><td>Да</td><td>Да</td><td>Да</td>
</tr>
</table>
</body>
</html>
```



Браузер	Internet Explorer		Opera			Firefox	
	6.0	7.0	7.0	8.0	9.0	1.0	2.0
Поддерживается	Нет	Да	Нет	Да	Да	Да	Да

В данной таблице установлено восемь колонок и три строки. Часть ячеек с надписями «Internet Explorer», «Opera» и «Firefox» объединены где по две, а где и по три ячейки. В ячейке с надписью «Браузер» применено объединение по вертикали.

### Вложенные таблицы

Объединение ячеек имеет некоторые недостатки, поэтому этот метод создания таблиц нельзя использовать повсеместно.

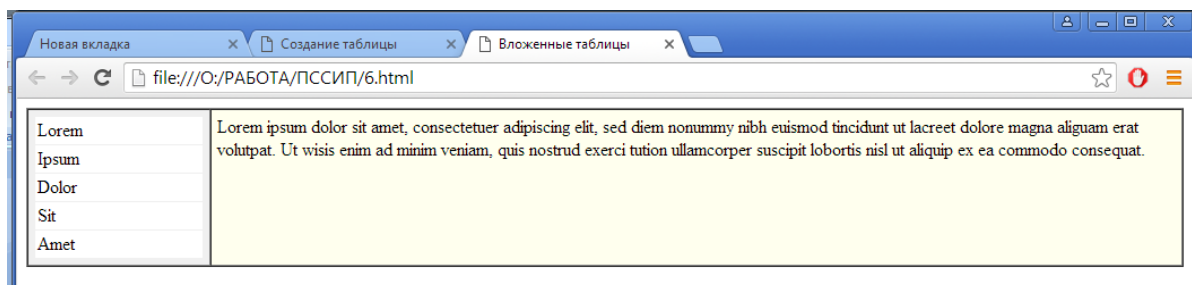
Для упрощения верстки применяется прием с вложенными таблицами.

Суть идеи проста — в ячейку вкладывается еще одна таблица со своими параметрами. Поскольку эти таблицы в каком-то смысле независимы, то можно создавать довольно причудливые конструкции. Чтобы вложенная таблица занимала всю ширину ячейки, таблице надо задать ширину 100%.

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Пример (вложенная таблица):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>Вложенные таблицы</title>
</head>
<body>
  <table width="100%" border="1" cellpadding="5" cellspacing="0">
    <tr>
      <td width="150" valign="top" bgcolor="#f0f0f0">
        <table width="100%" cellpadding="2" cellspacing="1">
          <tr><td bgcolor="#ffffff">Lorem</td></tr>
          <tr><td bgcolor="#ffffff">Ipsum</td></tr>
          <tr><td bgcolor="#ffffff">Dolor</td></tr>
          <tr><td bgcolor="#ffffff">Sit</td></tr>
          <tr><td bgcolor="#ffffff">Amet</td></tr>
        </table>
      </td>
      <td valign="top" bgcolor="#ffffee">Lorem ipsum dolor sit amet, consectetur
        adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet
        dolore magna aliquam erat volutpat. Ut wisis enim ad minim veniam, quis
        nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea
        commodo consequat.</td>
    </tr>
  </table>
</body>
</html>
```



В данном макете с помощью таблицы создается две колонки, причем левая колонка имеет фиксированную ширину 150 пикселей. Чтобы создать подобие навигации, внутри ячейки добавлена еще одна таблица с шириной 100%.

Если не задавать границы, то определить наличие таблиц по виду веб-страницы довольно сложно. По этой причине таблицы до сих пор активно применяются для верстки сложных макетов.

### Заголовок таблицы

При большом количестве таблиц на странице каждой из них удобно задать заголовок, содержащий название таблицы и ее описание. Для этой цели в HTML существует специальный тег **<caption>**, который устанавливает текст и его положение относительно таблицы. Тег **<caption>** находится внутри контейнера **<table>**, это его стандартное местоположение. Проще всего размещать текст по центру таблицы сверху или снизу от нее, в остальных случаях браузеры по-разному интерпретируют атрибуты тега **<caption>**, из-за чего результат получается неодинаковый. По умолчанию заголовок помещается сверху

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.

Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты  
таблицы по центру, его ширина не превышает ширины таблицы и в случае длинного текста он автоматически переносится на новую строку. Для изменения положения заголовка у тега <caption> существует атрибут **align**, который может принимать следующие значения:

- **left** — выравнивает заголовок по левому краю таблицы. Браузер Firefox располагает текст сбоку от таблицы, Internet Explorer и Opera располагают заголовок сверху, выравнивая его по левому краю;

- **right** — в браузере Internet Explorer и Opera располагает заголовок сверху таблицы и выравнивает его по правому краю таблицы. Firefox отображает заголовок справа от таблицы;

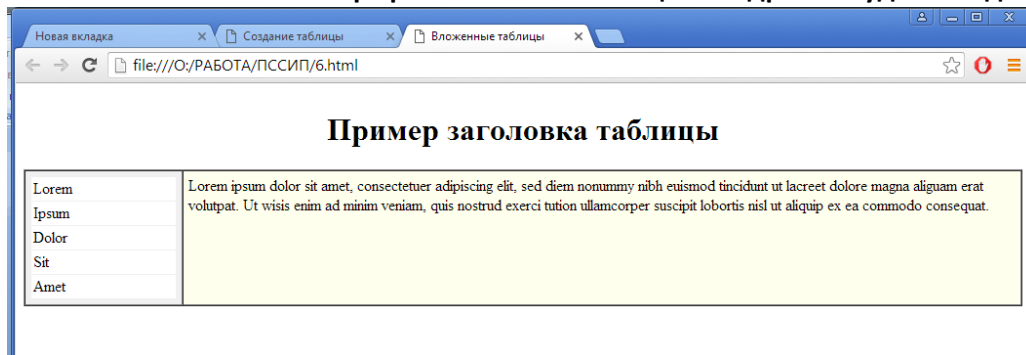
- **center** — заголовок располагается сверху таблицы по ее центру. Такое расположение задано в браузерах по умолчанию;

- **top** — результат аналогичен действию атрибута center, но в отличие от него входит в спецификацию HTML 4 и понимается всеми браузерами;

- **bottom** — заголовок размещается внизу таблицы по ее центру.

Пример (создание заголовка таблицы):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>Вложенные таблицы</title>
</head>
<body>
<table width="100%" border="1" cellpadding="5" cellspacing="0">
<caption><h1>Пример заголовка таблицы</h1></caption>
<tr>
<td width="150" valign="top" bgcolor="#f0f0f0">
<table width="100%" cellpadding="2" cellspacing="1">
<tr><td bgcolor="#ffffff">Lorem</td></tr>
<tr><td bgcolor="#ffffff">Ipsum</td></tr>
<tr><td bgcolor="#ffffff">Dolor</td></tr>
<tr><td bgcolor="#ffffff">Sit</td></tr>
<tr><td bgcolor="#ffffff">Amet</td></tr>
</table>
</td>
<td valign="top" bgcolor="#ffffee">Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet
dolore magna aliquam erat volutpat. Ut wisis enim ad minim veniam, quis
nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea
commodo consequat.</td>
</tr>
</table>
</body>
</html>
```



Таблицы являются одной из основных структур, используемых для структурирования информации в HTML-документах. Кроме того, таблицы часто используются для организации структуры страницы, и хотя сейчас такое использование таблиц признано устаревшим и не рекомендуемым, оно до сих пор применяется многими веб-дизайнерами.

Таблица-контейнер

Вложенная таблица					
		Таблица, вложенная вовложенную			

**Пример :**

[illegible][illegible]



```
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
```

```
</tr>
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
</table>
```

## Слои

Основная особенность слоев и их основное отличие в использовании от других способов верстки — точное позиционирование и способность накладываться друг на друга. Благодаря этой особенности с помощью слоев можно создавать разные эффекты на веб-странице.

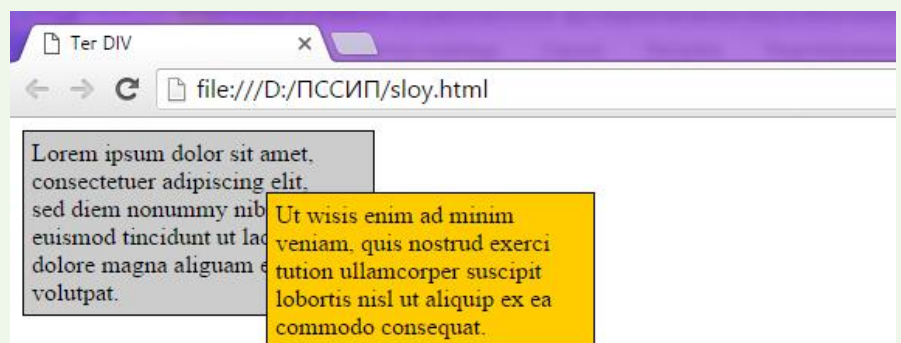
### Тег <div>

Элемент **<div>...</div>** является блочным элементом и предназначен для выделения фрагмента документа с целью изменения вида содержимого. Как правило, вид блока управляется с помощью стилей. Чтобы не описывать каждый раз стиль внутри тега, можно выделить стиль во внешнюю таблицу стилей, а для тега добавить атрибут **class** или **id** с именем селектора.

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

- **align** задает выравнивание содержимого тега **<div>**;
- **title** добавляет всплывающую подсказку к содержимому.

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset=" windows-1251 ">
<title>Ter DIV</title>
<style type="text/css">
.block1 {
width: 200px;
background: #ccc;
padding: 5px;
padding-right: 20px;
border: solid 1px black;
float: left;
}
.block2 {
width: 200px;
background: #fc0;
padding: 5px;
border: solid 1px black;
float: left;
position: relative;
```



```
top: 40px;
left: -70px;
}
</style>
</head>
<body>

<div class="block1">Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet
dolore magna aliquam erat volutpat.</div>
<div class="block2">Ut wisis enim ad minim veniam, quis nostrud
exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex
ea commodo consequat.</div>

</body>
</html>
```

Абсолютное позиционирование - Самый простой и менее гибкий способ создания наложения.

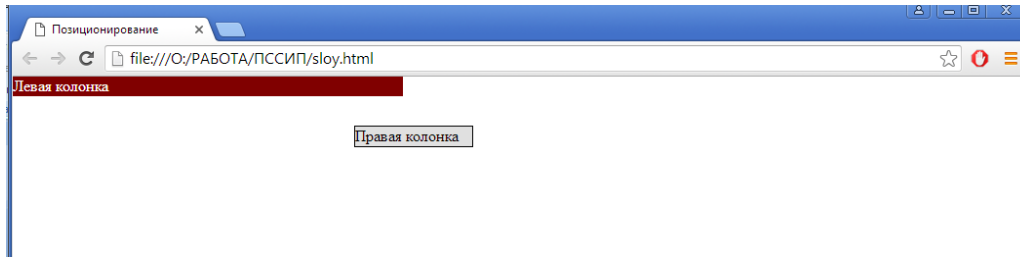
Абсолютное позиционирование позволяет накладывать слои в любом порядке друг на друга. Но при этом требуется знать точные координаты каждого слоя относительно одного из углов окна браузера, что не всегда возможно. *Поэтому данный подход имеет ограниченную область применения, например, для создания верхнего меню, когда его положение не меняется и точно зафиксировано.*

При использовании наложения требуется присвоить свойству **position** значение **absolute**. Само положение слоя регулируется свойствами **left**, **top**, **right** и **bottom**, которые задают координаты соответственно от левого, верхнего, правого и нижнего края.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>Позиционирование</title>
<style type="text/css">
#rightcol {
position: absolute; /* Абсолютное позиционирование */
left: 350px; /* Положение левого края */
top: 50px; /* Положение верхнего края */
width: 120px; /* Ширина слоя */
background: #e0e0e0; /* Цвет фона */
border: solid 1px #000; /* Параметры рамки */
}
#leftcol {
position: absolute; /* Абсолютное позиционирование */
left: 0; /* Положение левого края */
top: 0; /* Положение верхнего края */
width: 400px; /* Ширина слоя */
background: #800000; /* Цвет фона */
color: white; /* Цвет текста */
}
</style>
<body>
<div id="leftcol">Левая колонка</div>
```

```
<div id="rightcol">Правая колонка</div>  
</body>  
</html>
```



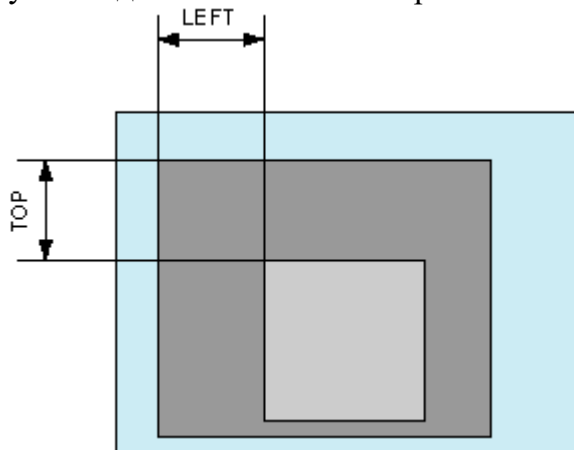
В данном примере положение слоя с именем **leftcol** устанавливается в левом верхнем углу окна браузера, а слой **rightcol** смещен на 350 пикселей вправо от левого края окна и на 50 пикселей вниз. Заметьте, что значения **left** и **top** следует указывать обязательно для всех слоев, чтобы получилось нужное наложение с заданными координатами.

Поскольку вывод содержимого слоя осуществляется в заданное место, то порядок описания слоев указывает и порядок их наложения друг на друга. Самый первый слой, приведенный в коде веб-страницы, будет располагаться на заднем плане, а самый последний — на переднем. Порядок также можно менять с помощью свойства **z-index**. Чем больше его значение, тем выше располагается текущий слой относительно других слоев.

Более интересный подход к созданию наложения — использование относительного позиционирования. В этом случае слои можно размещать по центру окна браузера или располагать их в любом месте веб-страницы, не задумываясь уже над значением координат слоев.

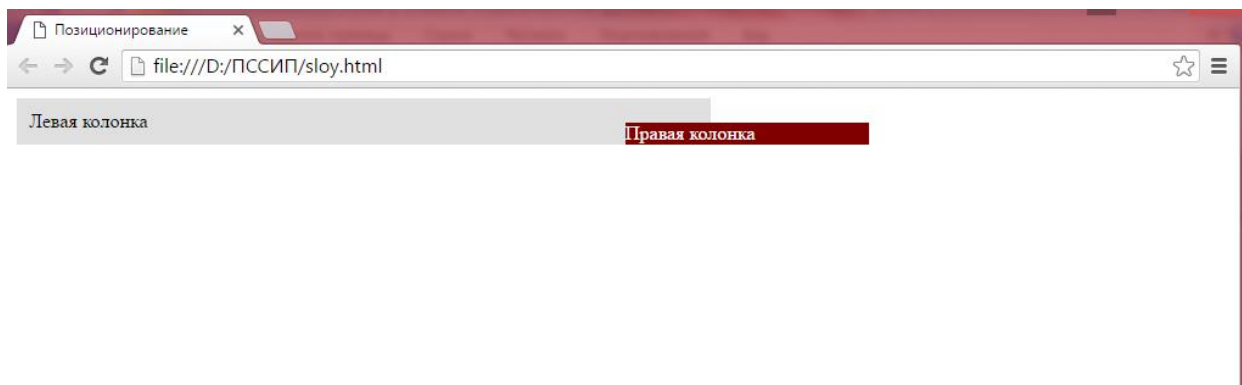
#### Относительное позиционирование

Чтобы наложить один слой на другой и не привязываться жестко к координатной сетке, можно попробовать следующий подход. Для первого слоя, который будет располагаться на заднем плане, указываем абсолютное позиционирование, присваивая свойству **position** значение **absolute**. Второй слой, расположенный поверх первого, должен иметь относительное позиционирование, что достигается с помощью значения **relative** у свойства **position**. Положение верхнего слоя определяется от левого верхнего угла нижнего слоя заданием **left** и **top** (на рисунке задание положения верхнего слоя).



Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset="windows-1251">
<title>Позиционирование</title>
<style type="text/css">
#leftcol {
position: absolute; /* Абсолютное позиционирование */
width: 550px; /* Ширина левой колонки */
background: #e0e0e0; /* Цвет фона содержимого */
padding: 10px /* Поля вокруг текста */
}
#rightcol { /* Этот слой накладывается поверх */
position: relative; /* Относительное позиционирование */
left: 500px; /* Положение от левого края */
top: 20px; /* Положение от верхнего края */
width: 200px; /* Ширина правой колонки */
background: #800000; /* Цвет фона */
color: #fff; /* Цвет текста */
}
</style>
</head>
<body>
<div id="leftcol">Левая колонка</div>
<div id="rightcol">Правая колонка</div>
</body>
</html>
```



*В примере ширина слоев задается свойством width, а местоположение верхнего слоя (он называется rightcol) свойствами left и top. Как указывалось выше, порядок наложения слоев определяется их порядком описания в коде или с помощью z-index. Поэтому слой с именем leftcol будет располагаться на заднем плане, поскольку он определен самым первым.*

Существует и другой способ наложения слоев, который связан с относительным позиционированием и использует все разнообразие средств размещения разных слоев.

### Универсальный подход

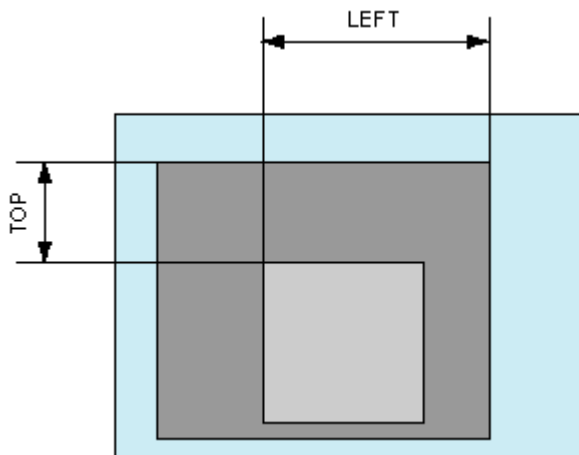
Теоретически, можно разместить слои по вертикали друг под другом и поднять нижний слой вверх с помощью свойства top, задавая ему отрицательное значение или с помощью bottom. На практике добиться подобного довольно сложно, ведь определить высоту слоя простыми средствами, а, следовательно, и величину, на которую следует смещать слой, не представляется возможным, поскольку она зависит от размера шрифта, содержимого слоя и многих других параметров. Проще отсчет координат вести от верхнего

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.**

**Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**  
угла одного из слоев. Для этого следует разместить слои рядом по вертикали, а потом уже смещать один слой относительно другого.

Реализуется это следующим способом. У каждого слоя необходимо указать конструкцию `float: left`, которая позволяет один слой пристыковать к другому справа. Добавлять `float` следует для каждого слоя, иначе в некоторых браузерах появится промежуток между слоями.

Теперь слои располагаются рядом, и прежде, чем указывать координаты, задаем относительное позиционирование значением `relative` у свойства `position`. Положение верхнего слоя управляется значением `left` и `top`. Но поскольку отсчет координат в данном случае ведется от левого верхнего угла второго слоя, по горизонтали нужно указывать отрицательное значение (рисунок задание положения верхнего слоя). Впрочем, можно использовать также свойство `bottom`.

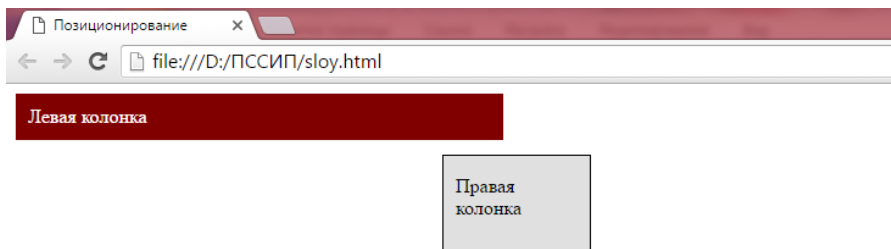


Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset= windows-1251">
<title>Позиционирование</title>
<style type="text/css">
#leftcol {
position: relative; /* Относительное позиционирование */
float: left; /* Совмещение колонок по горизонтали */
width: 400px; /* Ширина слоя */
background: #800000; /* Цвет фона */
color: white; /* Цвет текста */
}
#rightcol {
position: relative; /* Относительное позиционирование */
float: left; /* Совмещение колонок по горизонтали */
left: -50px; /* Сдвиг слоя влево */
top: 50px; /* Смещение слоя вниз */
width: 120px; /* Ширина слоя */
background: #e0e0e0; /* Цвет фона */
border: solid 1px black; /* Параметры рамки */
}
#leftcol P {
padding: 10px; /* Поля вокруг текста */
padding-right: 50px; /* Значение поля справа */
margin: 0; /* Обнуляем значения отступов */
}
</style>
</head>
<body>
<div id="leftcol">
<div id="rightcol">
<div id="rightcol">
</div>
</div>
</body>
</html>
```

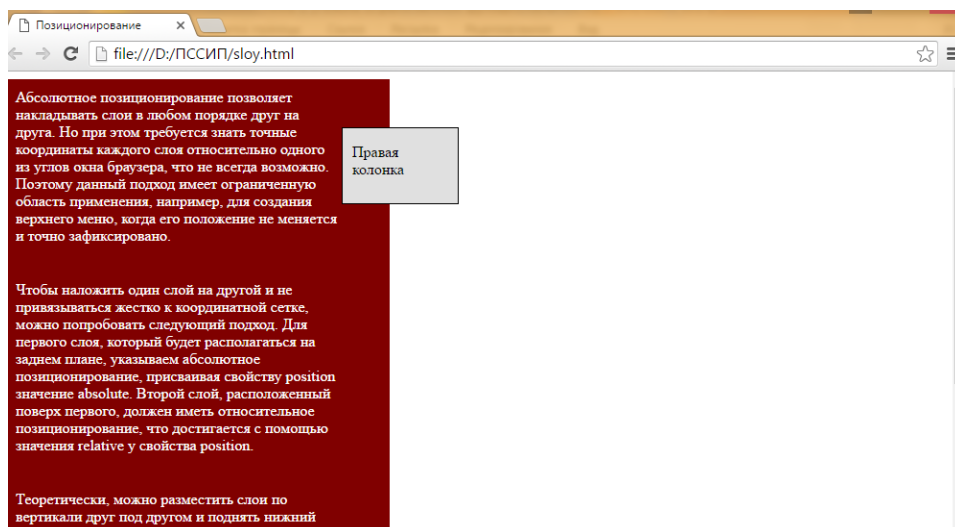
Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

```
#rightcol P {  
  padding: 10px; /* Поля вокруг текста */  
  padding-top: 0 /* Значение поля сверху */  
}  
</style>  
</head>  
<body>  
  <div id="leftcol">  
    <p>Левая колонка </p>  
  </div>  
  <div id="rightcol">  
    <p>Правая колонка</p>  
  </div>  
</body>  
</html>
```



В примере верхний слой с именем `rightcol` смещается на 50 пикселей по горизонтали и вертикали. Чтобы он не закрывал при наложении содержимое слоя `leftcol`, справа у текста делается отступ через свойство `padding-right`.

Как видно из примера, для создания наложения слоев требуется задать всего два свойства у нижнего слоя и четыре у верхнего. Остальные стилевые свойства управляют видом самих слоев и их содержимого.



## Карты-изображения

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

*Карты-изображения позволяют привязывать ссылки к разным областям одного изображения. Реализуется в двух различных вариантах — серверном и клиентском. В случае применения серверного варианта браузер посылает запрос на сервер для получения адреса выбранной ссылки и ждет ответа с требуемой информацией. Такой подход требует дополнительного времени на ожидание результата и отдельные файлы для каждой карты-изображения.*

*В клиентском варианте карта располагается в том же HTML-документе, что и ссылка на изображение.*

Карты-изображения позволяют привязывать ссылки к разным областям одного изображения - это способ сделать различные части одного графического изображения гиперссылками. Они позволяют выделить отдельные области изображений и определить для каждой из них свое действие.

#### Клиентский вариант карты-изображения

Для указания того, что изображение является картой, используется атрибут **usemap** тега **<img>**. В качестве значения указывается ссылка на описание конфигурации карты.



*На рисунке закладки являются ссылками, созданными с помощью карты*

*Чтобы сделать ссылки на закладки, показанные на рисунке, следует использовать следующий код.*

Для указания браузеру, что изображение является картой, применяется атрибут **usemap**. Он является ссылкой на описание конфигурации карты, которая задается тегом **<map>**. Значение атрибута **name** данного тега должно соответствовать имени в **usemap**. Для задания активной области, являющейся ссылкой на HTML-документ, используется тег **<area>** (значение параметра **name** тега **<map>** должно соответствовать имени в **usemap**).

#### Пример:

```
<map name="Map">
```

```
</map>
```

#### Атрибуты тега AREA:

**shape** - определяет форму активной области. Форма может быть в виде окружности (circle), прямоугольника (rect - по умолчанию), полигона (poly);

**alt** - добавляет альтернативный текст для каждой области. Служит лишь комментарием для ссылки, поскольку на экран не выводится;

**coords** - задает координаты активной области. Координаты отсчитываются в пикселах от левого верхнего угла изображения, которому соответствует значение 0,0. Первое число является координатой по горизонтали, второе — по вертикали. Список координат зависит от формы области.

Для окружности задаются три числа — координаты центра круга и радиус.

```
<area shape="circle" coords="230,340, 100" href="circle.html">  
- // -
```



Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.

Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

```
<area shape="circle" coords="133,116,59" alt="Нет ссылки (nonhref)" nonhref>
```

Для прямоугольника — координаты левого верхнего и правого нижнего угла.

```
<area shape="rect" coords="24,18, 210,56" href="rect.html">
```

- // -

```
<area shape="rect" coords="-1,1,264,232" href="fromimagemap2.htm" alt="На Документ 2">
```

Для полигона задаются координаты его вершин.



**href** - определяет адрес ссылки для области. Правила записи такие же, как и для тега <a>.

```
coords="355,75,400,99,450,130,363,295,356,313,348,348,181,291,189,  
281,260,291,305,290,316,248,317,224,318,197" href="fromimagemap3.htm" alt="На Документ  
3">
```

Назначение атрибутов **href**, **target** и **alt** такое же, как и в случае тега <a>.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">  
<html>  
  <head>  
    <meta http-equiv="Content-Type" content="text/html; charset= windows-1251">  
    <title>Карта-изображение</title>  
  </head>  
  <body>  
  
    <p></p>  
    <p><map name="map"> <area shape="poly" alt="Закладка 2"  
      coords="210,27, 203,9, 202,6, 197,2, 192,1, 120,1, 115,2, 110,6, 112,9, 119,27,  
119,32, 211,32, 210,27"  
      href="2.html">  
      <area shape="poly" alt="Закладка 3"  
        coords="302,27, 295,9, 293,6, 289,2, 283,1, 212,1, 206,2, 202,6, 203,9, 210,27,  
211,32, 284,32, 303,32, 302,27" href="3.html">  
      <area shape="poly" alt="Закладка 4"  
        coords="302,27, 303,32, 394,32, 393,27, 386,9, 382,3, 375,1, 303,1, 298,2, 293,6,  
295,9, 302,27"  
        href="4.html">  
    </map></p>  
  </body>  
</html>
```

Атрибут **nonhref** используется для того, чтобы указать, что область не является активной. Например, его можно использовать, чтобы сделать активной область в виде кольца.

```
  
<map name="Map2">  
<area shape="circle" coords="133,116,59" alt="Нет ссылки (nonhref)" nonhref>
```

```
<area shape="circle" coords="133,117,89" href="fromimagemap1.htm" alt="На Документ 1">
</map>
```

Карты-изображения позволяют создавать ссылки на разные области одного изображения. *Использование этого подхода наглядней, чем обычные текстовые ссылки и позволяет применять всего один графический файл для организации ссылок. Однако не нужно считать, что карты-изображения следует включать везде, где требуются графические ссылки. Прежде всего, следует оценить все доводы за и против, а также просмотреть альтернативные варианты.*

#### Преимущества карт-изображений:

1. Карты позволяют задать любую форму области ссылки. Учитывая, что изображения по своей природе прямоугольны, сделать графическую ссылку сложной формы, например для указания географического района, без карт-изображений не представляется возможным. Как правило, в географической тематике карты-изображения и применяются наиболее часто.

2. С одним файлом удобней работать — не приходится заботиться о состыковке отдельных фрагментов и рисунок легко можно поместить в нужное место.

#### Недостатки:

1. Нельзя установить всплывающую подсказку и альтернативный текст для отдельных областей. Альтернативный текст позволяет получить текстовую информацию о рисунке при отключенной в браузере загрузке изображений. Поскольку загрузка изображений происходит после получения браузером информации о нем, то замещающий рисунок текст появляется раньше. А уже по мере загрузки текст будет сменяться изображением. Для карт-изображений эта особенность является актуальной, ведь если отключить просмотр изображений, что делают многие пользователи, то в итоге увидим лишь один пустой прямоугольник.

2. При сложной форме области ссылки увеличивается объем кода HTML. Контур аппроксимируется набором прямых отрезков, для каждой точки такого отрезка следует задать две координаты, а общее количество таких точек может быть достаточно велико. Справедливости ради, следует отметить, что сложные формы являются частным случаем и применяются достаточно редко.

3. С картами-изображениями нельзя сделать разные эффекты, которые доступны при разрезании одного рисунка на фрагменты: эффект перекашивания, частичная анимация, индивидуальная оптимизация картинок для их быстрой загрузки.

#### Юзабилити:

С позиции удобства пользователей, карты-изображения имеют только одно преимущество — ссылки разнообразной формы. Это добавляет наглядность в представлении информации — мы не ограничены прямоугольной формой ссылки и можем использовать ссылки сложной конфигурации для своих целей. Во всех остальных отношениях от них проку нет — обычные текстовые ссылки более информативны и им не страшно отключение показа картинок в браузере. Тот факт, что одно изображение загружается быстрее, чем та же картинка, но порезанная на фрагменты и сохраненная в виде набора графических файлов, легко обходится. Каждый из таких конечных файлов можно уменьшить, используя индивидуальные настройки оптимизации. В итоге, общий объем всех фрагментов будет занимать меньше места, чем одно изображение. Не стоит сбрасывать со счетов и психологический фактор — человеку кажется, что набор маленьких картинок загружается быстрее, чем одна большая.

Основной недочет карт — нет четко выделенных границ ссылок. Поэтому эти границы приходится выделять разными средствами уже на изображении. Если рисунок не загрузился

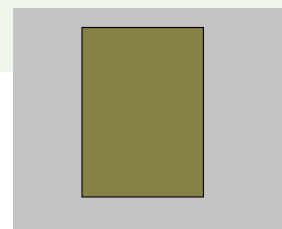
Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты по каким-либо причинам, то разобраться в наборе ссылок становится весьма проблематичным.

### Карты-изображения и их исходные коды

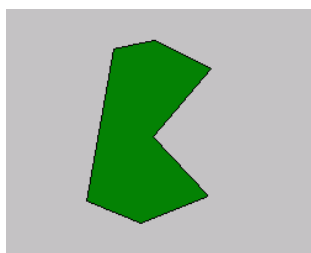
#### Прямоугольник:

```

<map id="Прямоугольник" name="Прямоугольник">
<area shape="rect" coords="63,18,172,168" href="А:\HTML\Навигационн
ые карты.пример.html" title="Прямоугольник" target="blank" >
<area shape="default" nohref="nohref" alt="" >
</map>
```



#### Многоугольник:



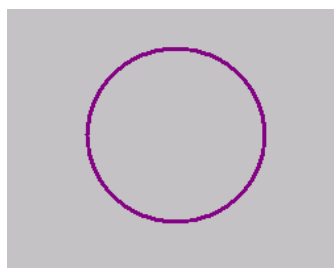
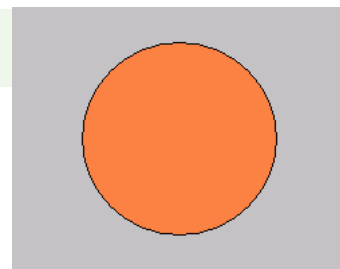
```

<map id="Многоугольник" name="Многоугольник">
<area shape="poly" coords="65,153,87,32,120,25,166,48,119,103,164
,151,110,174" href="Навигационные%20карты.пример.html" title="Мно
гоугольник" target="blank" >
<area shape="default" nohref="nohref" alt="" >
</map>
```

#### Круг:

```

<area shape="circle" coords="126,99,73"
href="Навигационные%20карты.
пример.html" title="круг" target="blank" >
<area shape="default" nohref="nohref" alt="" >
</map>
```



#### Кольцо:

```

<map id="Кольцо" name="Кольцо">
<area shape="circle" coords="128,96,66" nohref="nohref"
title="кольцо">
<area shape="circle" coords="128,96,70"
href="Навигационные%20карты.
пример.html" title="" target="blank">
<area shape="default" nohref="nohref" alt="">
</map>
```

#### Пример:

```

<map name="Map">
<area shape="poly" cords="4,43,31,45,162,10,198,9,205,21,211,
63,221,204,233,258,199,268,132,275,72,290,48,249,33,149" href="#" alt="На Документ 1">
<area shape="poly" coords="262,20,270,20" href="#">
<area shape="poly" cords="215,18,268,18,350,38,313,186,300,
282,257,282,196,271,240,261,234,199,221,133,220,89" href="fromimagemap2.htm" alt="На
Документ 2">
<area shape="poly" coords="355,75,400,99,450,130,363,295,356,
```

```
313,348,348,181,291,189,281,260,291,305,290,316,  
248,317,224,318,197"  
href="fromimagemap3.htm" alt="На Документ 3">  
</map>
```

## Фреймы

Фрейм (от англ. frame — рамка) — отдельный, законченный HTML-документ, который вместе с другими HTML-документами может быть отображён в окне браузера.

*Фреймы по своей сути очень похожи на ячейки таблицы, однако более универсальны. Фреймы разбивают веб-страницу на отдельные миникадры, расположенные на одном экране, которые являются независимыми друг от друга. Каждое окно может иметь собственный адрес. При нажатии на любую из ссылок, расположенных в одном фрейме, можно продолжать видеть страницы в других окнах.*

### Создание фреймов

Для создания фрейма используется тег `<frameset>`, который заменяет тег `<body>` в документе и применяется для разделения экрана на области. Внутри данного тега находятся теги `<frame>`, которые указывают на HTML-документ, предназначенный для загрузки в область.

### Просто пример фреймовой структуры:

```
<html>  
  <head>  
    <title>Фреймы в html</title>  
  </head>  
  <frameset rows="30%, 10%, 60%" >  
    <frame>  
    <frame>  
    <frame>  
  </frameset>  
</html>
```

Тег `<frame>` определяет свойства отдельного фрейма, на которые делится окно браузера. Этот элемент должен располагаться в контейнере `<frameset>`, который к тому же задает способ разметки страницы на отдельные области. В каждую из таких областей загружается самостоятельная веб-страница определяемая с помощью атрибута `src`. Хотя обязательных атрибутов у тега `<frame>` и нет, рекомендуется задавать каждому фрейму его имя через атрибут `name`. Это особенно важно, если требуется по ссылке из одного фрейма загружать документ в другой.

### Атрибуты:

**bordercolor** - цвет линии границы;

**frameborder** - отображать рамку вокруг фрейма или нет;

**name** - задает уникальное имя фрейма;

**noresize** - определяет, можно изменять размер фрейма пользователю или нет;

**scrolling** - способ отображения полосы прокрутки во фрейме;

**src** - путь к файлу, предназначенному для загрузки во фрейме;

**marginwidth** - определяет отступ во фрейме от левого края до содержимого.

**marginheight** - определяет отступ во фрейме от верхнего края до содержимого;

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.

Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

**scrolling** - если содержимое фрейма не помещается в окно, то будут появляться полосы прокрутки, иногда это нарушает дизайн. Этот параметр помогает управлять отображением полос прокрутки. Может принимать значения: yes, no, auto;

**noresize** - если установить курсор мыши на рамки фрейма, то можно выполнить его перемещение. Для предотвращения этой возможности и используется этот параметр.

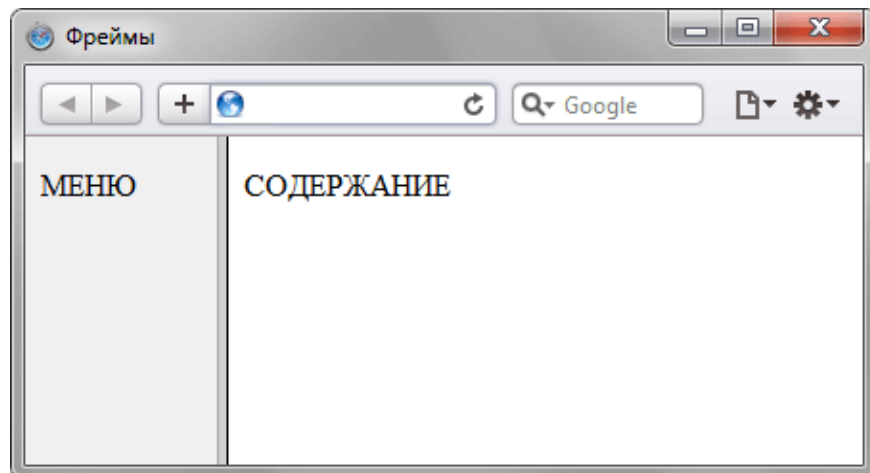
Закрывающий тег не требуется.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Ter FRAME</title>
</head>

<frameset rows="80,*" cols="*">
  <frame src="top.html" name="topFrame" scrolling="no" noresize>
  <frameset cols="80,*">
    <frame src="left.html" name="leftFrame" scrolling="no" noresize>
    <frame src="main.html" name="mainFrame">
  </frameset>
</frameset>

</html>
```



Пример разделения окна браузера на два фрейма

При использовании фреймов необходимо как минимум три HTML-файла: первый определяет фреймовую структуру и делит окно браузера на две части, а оставшиеся два документа загружаются в заданные окна. *Количество фреймов не обязательно равно двум, может быть и больше, но никак не меньше двух, иначе вообще теряется смысл применения фреймов.*

Рассмотрим этапы создания фреймов на основе страницы, продемонстрированной на рисунке.

Нам понадобится три файла:

- 1) **index.html** — определяет структуру документа;
- 2) **menu.html** — загружается в левый фрейм;

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.  
Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

3) **content.html** — загружается в правый фрейм.

Из них только index.html отличается по структуре своего кода от других файлов.

*Пример (файл index.html):*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>Использование фреймов</title>
</head>
<frameset cols="200,*">
  <frame src="menu.html" name="MENU">
  <frame src="content.html" name="CONTENT">
</frameset>
</html>
```

```
<frameset cols="1*, 2*" > // второй фрейм будет шире первого в 2 раза
```

В случае использования фреймов в первой строке кода пишется следующий тип документа.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd"
```

Данный **<!DOCTYPE>** указывает браузеру, что он имеет дело с фреймами, эта строка кода является обязательной. Контейнер **<head>** содержит типовую информацию вроде кодировки страницы и заголовка документа. Заголовок остается неизменным, пока HTML-файлы открываются внутри фреймов.

В данном примере окно браузера разбивается на две колонки с помощью атрибута **cols**, левая колонка занимает 100 пикселей, а правая — оставшееся пространство, заданное символом звездочки. Ширину или высоту фреймов можно также задавать в процентном отношении, наподобие таблиц.

В теге **<frame>** задается имя HTML-файла, загружаемого в указанную область с помощью атрибута **src**. В левое окно будет загружен файл, названный menu.html, а в правое — content.html. Каждому фрейму желательно задать его уникальное имя, чтобы документы можно было загружать в указанное окно с помощью атрибута name.

*Пример (файл menu.html):*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>меню для фрейма</title>
</head>
<body style="background: #f0f0f0">
  <p><h1>ПУТЬ К ГОМЕРУ</h1></p><br>
ПЕСНЬ ПЕРВАЯ<br>
<br>
ПЕСНЬ ВТОРАЯ<br><br>
```

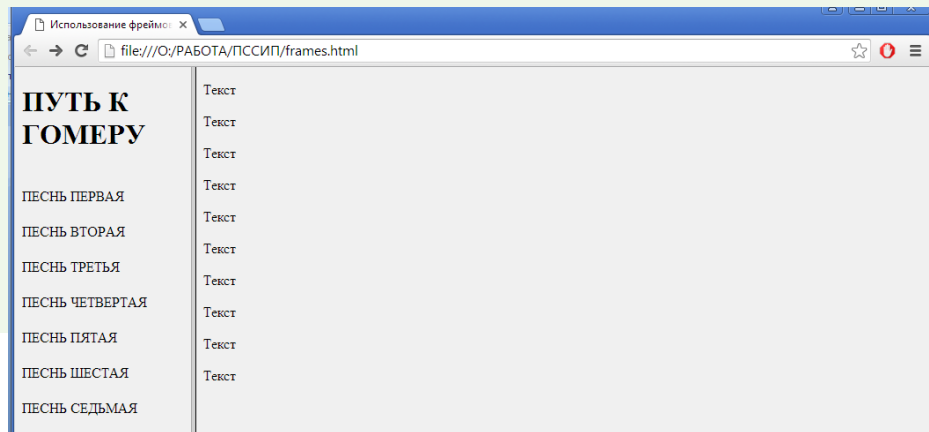
Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

```
ПЕСНЬ ТРЕТЬЯ<br><br>
ПЕСНЬ ЧЕТВЕРТАЯ<br><br>
ПЕСНЬ ПЯТАЯ<br><br>
ПЕСНЬ ШЕСТАЯ<br><br>
ПЕСНЬ СЕДЬМАЯ
</body>
</html>
```

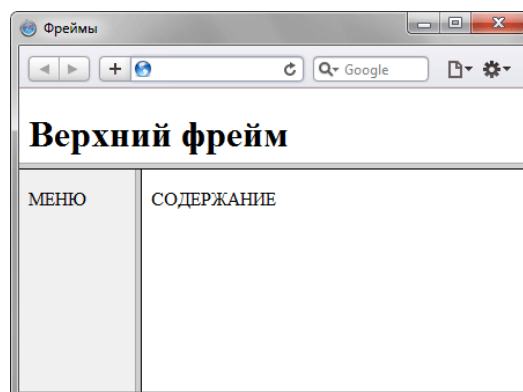
В данном примере серый фон на странице задается с помощью стилей.

Пример (файл content.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>меню для фрейма</title>
</head>
<body style="background: #f0f0f0">
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
</body>
</html>
```



Рассмотрим более сложный пример уже с тремя фреймами (разделение страницы на три фрейма).

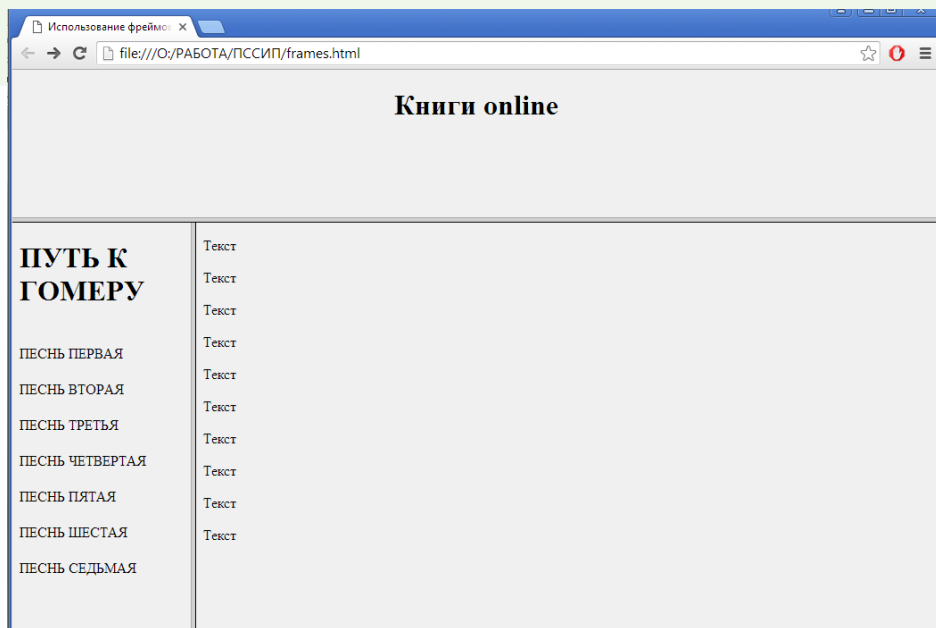


В данном случае опять используется тег `<frameset>`, но два раза, причем один тег вкладывается в другой. Горизонтальное разбиение создается через атрибут `rows`, где для разнообразия применяется процентная запись.

Пример (три фрейма):



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>Использование фреймов</title>
</head>
<frameset rows="25%,75%">
  <frame src="top.html" name="TOP" scrolling="no" noresize>
<frameset cols="200,*">
  <frame src="menu.html" name="MENU">
  <frame src="content.html" name="CONTENT">
</frameset>
</frameset>
</html>
```



Как видно из данного примера, контейнер `<frameset>` с атрибутом `rows` вначале создает два горизонтальных фрейма, но вместо второго фрейма подставляется еще один `<frameset>`, который повторяет уже известную вам структуру. Чтобы не появилась вертикальная полоса прокрутки, и пользователь не мог самостоятельно изменить размер верхнего фрейма, добавлены атрибуты `scrolling="no"` и `noresize`.

#### Изменение размеров фреймов

По умолчанию размеры фреймов можно изменять с помощью курсора мыши, наведя его на границу между фреймами. Для блокировки возможности изменения пользователем размера фреймов следует воспользоваться атрибутом `noresize` тега `<frame>`.

Для случая двух фреймов этот атрибут можно указать лишь в одном месте. Естественно, если у одного фрейма нельзя изменять размеры, то у близлежащего к нему размеры тоже меняться не будут.

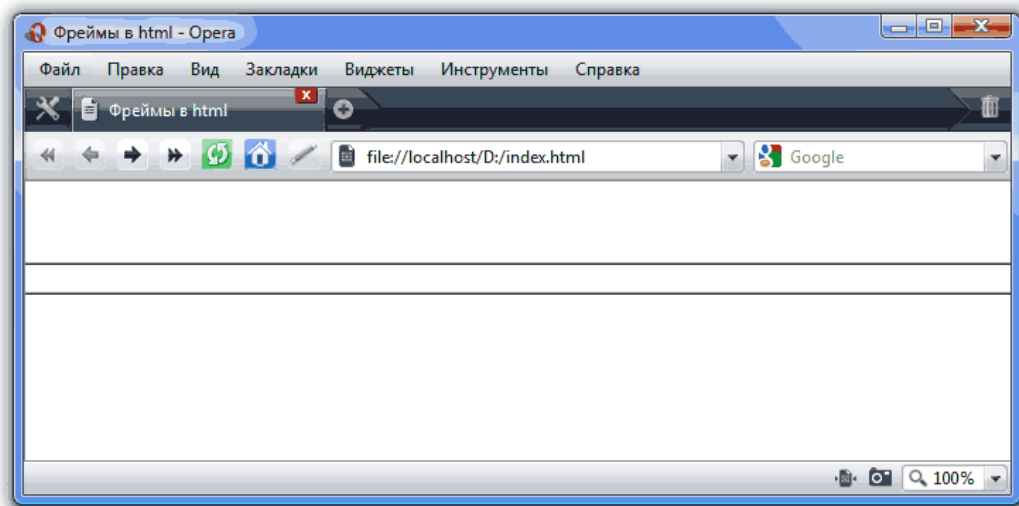
#### Полосы прокрутки

Если содержимое фрейма не помещается в отведенное окно, автоматически появляются полосы прокрутки для просмотра информации. В некоторых случаях полосы прокрутки нарушают дизайн веб-страницы, поэтому от них можно отказаться. Для управления отображением полос прокрутки используется атрибут `scrolling` тега `<frame>`. Он может принимать два основных значения: `yes` — всегда вызывает появление полос прокрутки, независимо от объема информации и `no` — запрещает их появление.

**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

*При выключенных полосах прокрутки, если информация не помещается в окно фрейма, просмотреть ее будет сложно. Поэтому `scrolling="no"` следует использовать осторожно.*

*Если атрибут `scrolling` не указан, то полосы прокрутки добавляются браузером только по необходимости, в том случае, когда содержимое фрейма превышает его видимую часть.*



*Просто пример фреймовой структуры (+ наполнение):  
top.html со следующим кодом:*

```
<html>
  <head>
    <title>Фреймы в html</title>
  </head>
  <body bgcolor="#FFCC66" text="#990000">
    <font size="5">шапка сайта</font>
  </body>
</html>
```

*menu.html со следующим кодом:*

```
<html>
  <head>
    <title>Меню сайта</title>
  </head>
  <body bgcolor="#FFCC66" text="#990000">
    <a>меню</a> <a>меню</a> <a>меню</a>
    <a>меню</a> <a>меню</a>
  </body>
</html>
```

*content.html с кодом:*

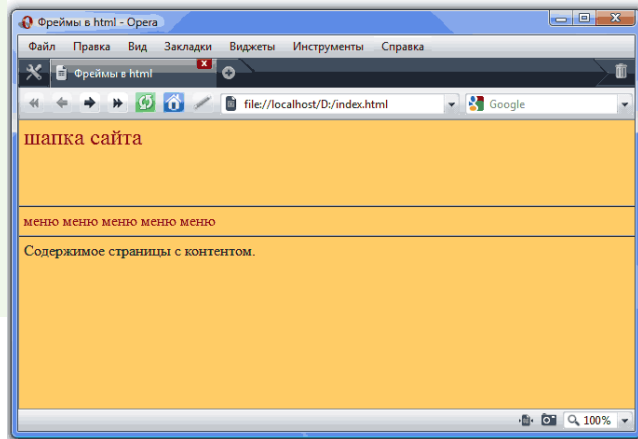
```
<html>
  <head>
    <title>Контент</title>
  </head>
  <body bgcolor="#FFCC66">
    Содержимое страницы с контентом.
  </body>
</html>
```

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

Эти страницы должны лежать в том же каталоге, что и index.html.

А теперь добавим параметр src в теги <frame> на нашей странице index.html:

```
<html>
<head>
  <title>Фреймы в html</title>
</head>
<frameset rows="30%, 10%, 60%" >
  <frame src="top.html">
  <frame src="menu.html">
  <frame src="content.html">
</frameset>
</html>
```



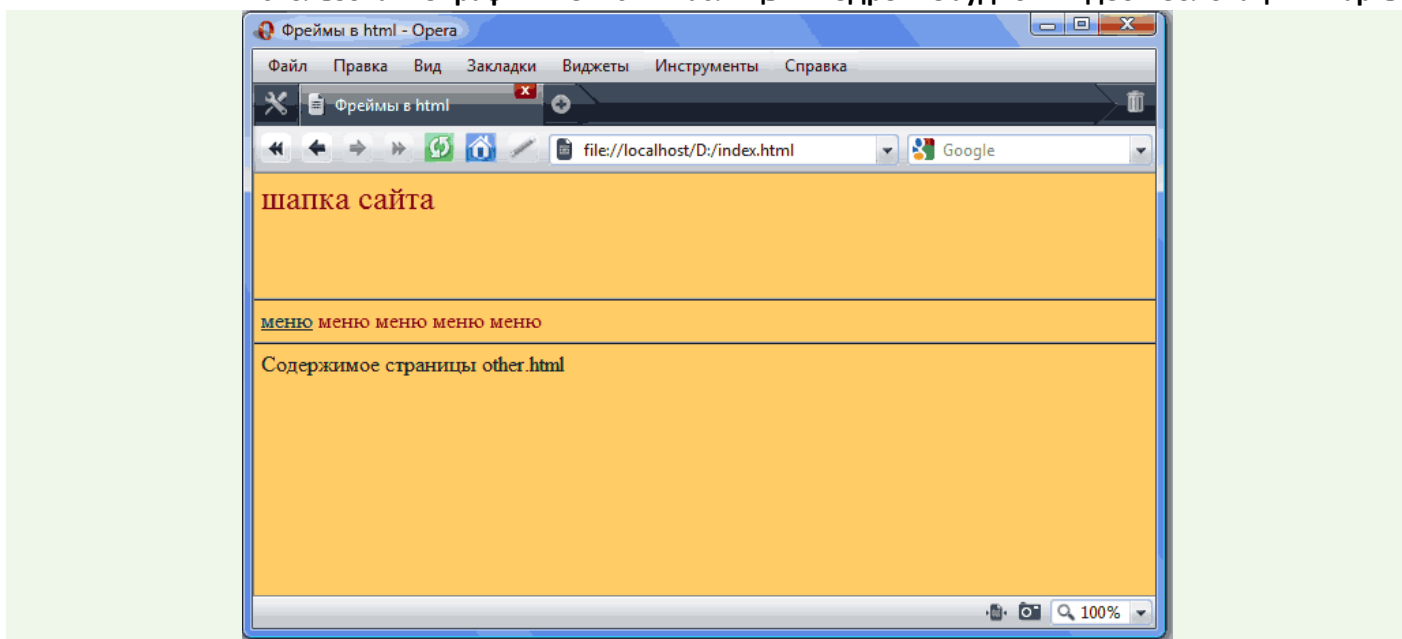
Теперь создайте еще одну страницу other.html с кодом:

```
<html>
<head>
  <title>Контент other.html</title>
</head>
<body bgcolor="#FFCC66">
  Содержимое страницы other.html
</body>
</html>
```

А теперь на странице menu.html сделаем первый пункт меню ссылкой на эту страницу и укажем, что открывать страницу other.html следует во фрейме с именем content:

```
<html>
<head>
  <title>Меню сайта</title>
</head>
<body bgcolor="#FFCC66" text="#990000">
  <a href="other.html" target="content">меню</a>
  <a>меню</a> <a>меню</a> <a>меню</a> <a>меню</a>
</body>
</html>
```

Теперь при щелчке по первому пункту меню, во фрейм content будет загружаться страница other.html



### Ссылки во фреймах

В обычном HTML-документе при переходе по ссылке, в окне браузера текущий документ заменяется новым. При использовании фреймов схема загрузки документов отличается от стандартной. Основное отличие — возможность загружать документ в выбранный фрейм из другого. Для этой цели используется атрибут **target** тега **<a>**. В качестве значения используется имя фрейма, в который будет загружаться документ, указанный атрибутом **name**.

Параметру **target** можно задать и другие значения:

**\_self** - загружать в тот же фрейм из которого ссылаются;

**\_top** - загружать в полное окно, закрывая остальные фреймы (обычно используется для ссылок на другие сайты).

### Пример:

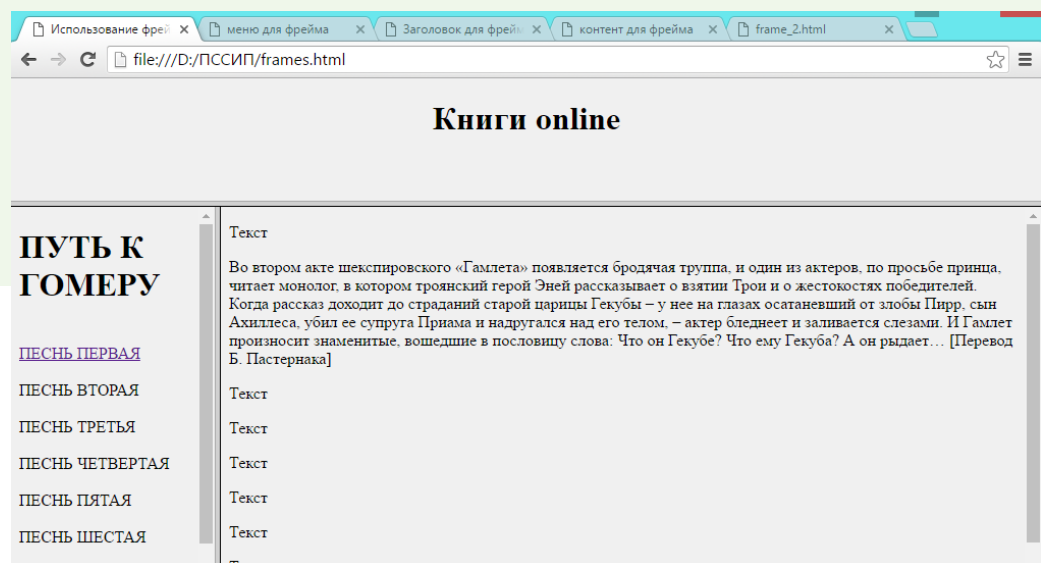
#### файл menu.html

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>меню для фрейма</title>
</head>
<body style="background: #f0f0f0">
  <p><h1>ПУТЬ К ГОМЕРУ</h1></p><br>
  <a href="frame_2.html" target="content">ПЕСНЬ ПЕРВАЯ</a>
<br>
<br>
ПЕСНЬ ВТОРАЯ<br><br>
ПЕСНЬ ТРЕТЬЯ<br><br>
ПЕСНЬ ЧЕТВЕРТАЯ<br><br>
ПЕСНЬ ПЯТАЯ<br><br>
ПЕСНЬ ШЕСТАЯ<br><br>
ПЕСНЬ СЕДЬМАЯ
</body>
</html>
```

#### файл frame\_2.html

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки.  
Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>контент для фрейма</title>
</head>
<body style="background: #f0f0f0">
<p>Текст</p>
<p>Во втором акте шекспировского «Гамлета» появляется бродячая труппа, и один из
актеров, по просьбе принца, читает монолог, в котором троянский герой Эней рассказывает
о взятии Трои и о жестокостях победителей. Когда рассказ доходит до страданий старой
царицы Гекубы – у нее на глазах осатаневший от злобы Пирр, сын Ахиллеса, убил ее
супруга Приама и надругался над его телом, – актер бледнеет и заливается слезами. И
Гамлет произносит знаменитые, вошедшие в пословицу слова:
Что он Гекубе? Что ему Гекуба?
А он рыдает... [Перевод В. Пастернака]</p>
<p></p>
<p></p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
<p>Текст</p>
</body>
</html>
```



### Границы между фреймами

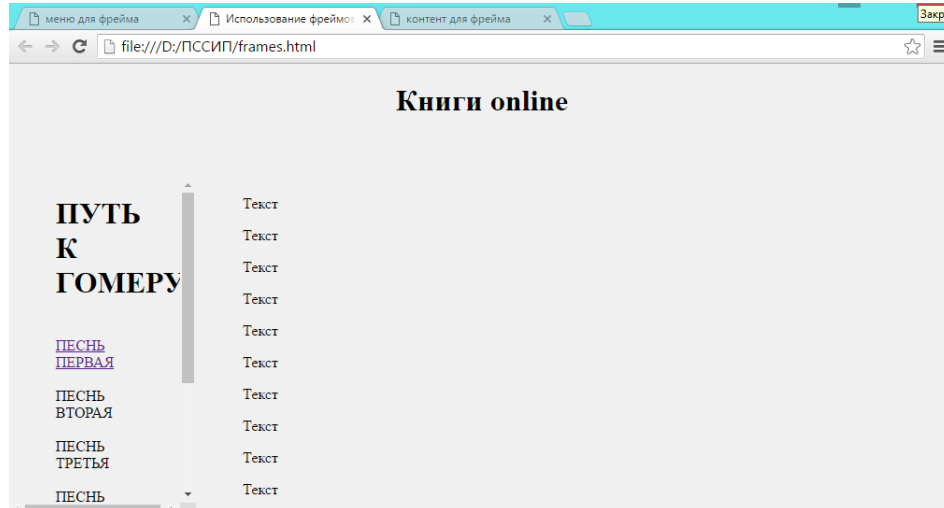
Граница между фреймами отображается по умолчанию и, как правило, в виде трехмерной линии. Чтобы ее скрыть используется атрибут `frameborder` тега `<frameset>` со значением 0. Однако в браузере Opera граница хоть и становится в этом случае бледной, все же остается. Для этого браузера требуется добавить `framespacing="0"`. Таким образом, комбинируя разные атрибуты тега `<frameset>`, получим универсальный код, который работает во всех браузерах. Линия при этом показываться никак не будет.

Внесем оставшиеся параметры тега `<frame>` в код нашей страницы `frames.html` и уберем рамки фреймов (для этого в тег `<frameset>` добавим два параметра `border="0"` `frameborder="0"`):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title>Использование фреймов</title>
</head>
<frameset rows="25%,75%" border="0" frameborder="0">
<frame src="top.html" name="TOP" scrolling="no" noresize>
```

Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты

```
<frameset cols="200,*">
  <frame src="menu.html" name="MENU" marginwidth="50">
  <frame src="content.html" name="content" marginwidth="50" marginheight="20">
</frameset>
</frameset>
</html>
```



Пример: убираем границу между фреймами

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Фреймы</title>
</head>
<frameset cols="100,*" frameborder="0" framespacing="0">
  <frame src="menu.html" name="MENU">
  <frame src="content.html" name="CONTENT">
</frameset>
</html>
```

Учтите, что атрибуты **frameborder** и **framespacing** не являются валидными и не соответствуют спецификации HTML.

Если граница между фреймами все же нужна, в браузере она рисуется по умолчанию, без задания каких-либо атрибутов. Можно, также, задать цвет рамки с помощью атрибута **bordercolor**, который может применяться в тегах **<frameset>** и **<frame>**. Цвет указывается по его названию или шестнадцатеричному значению, а толщина линии управляется атрибутом **border**. Браузер Opera игнорирует этот атрибут и обычно отображает линию черного цвета

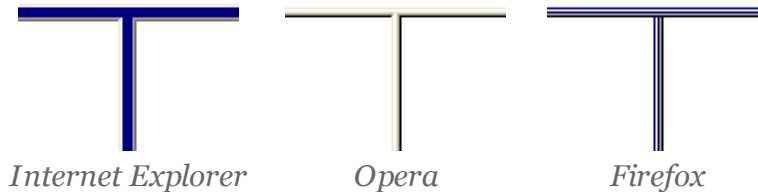
Пример: изменение цвета границы

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Фреймы</title>
</head>
<frameset cols="100,*" bordercolor="#000080" border="5">
```

```
<frame src="menu.html" name="MENU">  
<frame src="content.html" name="CONTENT">  
</frameset>  
</html>
```

Атрибуты **bordercolor** и **border** тега **<frameset>** также не являются валидными и не признаются спецификацией HTML.

В примере линия между фреймами задается синего цвета толщиной пять пикселей. Линии различаются по своему виду в разных браузерах, несмотря на одинаковые параметры.



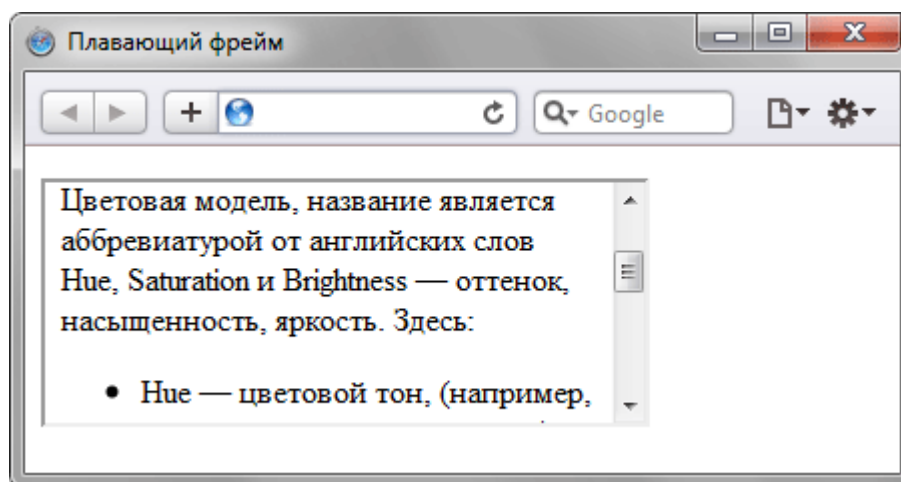
Браузер Opera никак не изменяет цвет границы между фреймами, Internet Explorer устанавливает широкую границу практически сплошного цвета, а Firefox границу отображает в виде набора линий.

### Плавающие фреймы

Разговор о фреймах будет неполным без упоминания плавающих фреймов. Так называется фрейм, который можно добавлять в любое место веб-страницы. Еще одно его название — встроенный фрейм, он называется так из-за своей особенности встраиваться прямо в тело веб-страницы.

Во фрейм можно загружать HTML-документ и прокручивать его содержимое независимо от остального материала на веб-странице. Размеры фрейма устанавливаются самостоятельно согласно дизайну сайта или собственных предпочтений.

Создание плавающего фрейма происходит с помощью тега **<iframe>**, он имеет обязательный атрибут **src**, указывающий на загружаемый во фрейм докумен.



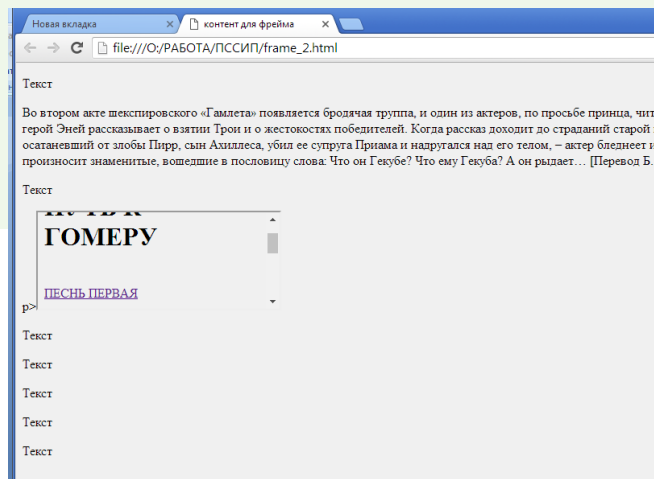
Пример использования тега **<iframe>**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```



**Единицы измерения в HTML. Физическое и логическое форматирование текста HTML-документа. Ссылки. Использование графики. Списки. Таблицы. Внедрение аудио и видео. Геолокация и карты**

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <title>контент для фрейма</title>
</head>
<body style="background: #f0f0f0">
  <p>Текст</p>
  <p>Во втором акте шекспировского «Гамлета» появляется бродячая труппа, и один из
актеров, по просьбе принца, читает монолог, в котором троянский герой Эней рассказывает
о взятии Трои и о жестокостях победителей. Когда рассказ доходит до страданий старой
царицы Гекубы – у нее на глазах осатаневший от злобы Пирр, сын Ахиллеса, убил ее
супруга Приама и надругался над его телом, – актер бледнеет и заливается слезами. И
Гамлет произносит знаменитые, вошедшие в пословицу слова: Что он Гекубе? Что ему
Гекуба? А он рыдает... [Перевод Б. Пастернака]</p>
  <p><iframe src="menu.html" width="300" height="120"></iframe></p>
  <p>Текст</p>
  <p>Текст</p>
  <p>Текст</p>
  <p>Текст</p>
  <p>Текст</p>
</body>
</html>
```



В данном примере ширина и высота фрейма устанавливается через атрибуты *width* и *height*. Сам загружаемый во фрейм файл называется *menu.html*. Заметьте, что если содержимое не помещается целиком в отведенную область, появляются полосы прокрутки.

Еще одно удобство плавающих фреймов состоит в том, что в него можно загружать документы по ссылке. Для этого требуется задать имя фрейма через атрибут *name*, а в теге *<a>* указать это же имя в атрибуте *target*.

*Пример (загрузка документа во фрейм)*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <title>Плавающий фрейм</title>
</head>
<body>
  <p><a href="rgb.html" target="color">RGB</a> |
    <a href="cmyk.html" target="color">CMYK</a> |
    <a href="hsb.html" target="color">HSB</a></p>
  <p><iframe src="model.html" name="color" width="100%" height="200"></iframe></p>
</body>
</html>
```

В данном примере добавлено несколько ссылок, они открываются во фрейме с именем *color*.

*Фреймы разделяют окно браузера на отдельные области, расположенные рядом друг с другом. В каждую из таких областей загружается самостоятельная веб-страница. Поскольку вокруг фреймов существует много разговоров об их необходимости, далее приведем достоинства и недостатки фреймов, чтобы можно было самостоятельно решить стоит ли их использовать на своем сайте.*

Плюсы и минусы фреймов:

Безусловным преимуществом является сокращение количества загружаемой на компьютер пользователя информации. Ведь шапка и меню сайта загружаются только один раз, а далее меняется только контент. Конечно, это сокращает время загрузки.

Но недостатков гораздо больше. Во-первых, в структуре фреймов легко запутаться.

Во-вторых, наше меню лежит в отдельном файле. А это значит, если пользователь нашел, например, вашу страницу content.html через поисковую систему, то он сможет прочитать только ее, ведь никаких ссылок и пунктов меню на этой странице нет.

И наконец, фреймовую структуру поддерживают не все браузеры. Поэтому существуют теги `<noframes>` `</noframes>`, они располагаются внутри тегов `<frameset>` `</frameset>` и содержат альтернативную информацию для браузеров не поддерживающих фреймы. По сути вам придется выполнить двойную работу и создать две версии сайта: с фреймами и без них.