

# Лабораторная работа № 1 Практическое создание web страницы с применением html5 и CSS3

**Цели занятия:** Научиться создавать файлы – интернет - страницы, содержащие элементы форматирования, написанные на языке HTML и CSS3.

## **Теоретический материал**

HyperTextMarkupLanguage (HTML) – это язык разметки документов во Всемирной паутине, принятый за стандартный. Большая доля всех Web-страниц в Интернете создана при помощи языка HTML (или XHTML), поэтому мы рассмотрим его подробно.

Язык HTML позволяет форматировать текст и другие элемента Web-страницы:

*Цвет, жирность, стиль, название шрифта* для текста.

*Позволяет выделять фрагменты и символы* например: ударение в слове, заголовок страницы или абзаца, сам абзац, пункт списка.

*Гипертекстовые ссылки*, позволяют переходить между документами и между фрагментами одного документа.

*Формы* для введения данных, как правило, данные из форм обрабатываются с помощью скриптов на языках программирования, ориентированных на Web, например PHP.

*Отображение мультимедийных файлов*, их может отображать сам браузер – изображения, аудиофайлы, или внешние приложения, взаимодействующие с браузером, например Flash-ролики, Java-апплеты.

HTML – язык разметки документов основанный на тэгах. Документ на языке HTML представляет собой набор элементов, при этом начало и конец каждого элемента обозначается служебными символами – тегами. Все тэги HTML начинаются с «<» (левой угловой скобки) и заканчиваются символом «>» (правой угловой скобки). Завершающий тег выглядит также, как начальный, и отличается от него прямым слэшем перед текстом внутри угловых скобок.

**<HTML></HTML>**

HTML регистронезависимый язык, теги могут быть написаны как строчными, так и заглавными буквами (в отличие от XHTML). Теги могут быть вложенными друг в друга.

**<HTML>**

**<HEAD>**

**<TITLE>**

Заголовок страницы

**</TITLE>**

**</HEAD>**

**<BODY>**

*Всё содержание документа – web страницы*

**</BODY>**

**</HTML>**

В общем виде синтаксис записи тега <BODY> со всеми допустимыми атрибутами выглядит так:

**<BODY BACKGROUND="URL" BGCOLOR="значение1" TEXT="значение2" LINK="значение3" VLINK="значение4" ALINK="значение5">**

тело документа HTML

<BODY>

Атрибут BACKGROUND позволяет дизайнеру поместить на web-страницу некий фоновый рисунок, записав в качестве параметра атрибута URL этого рисунка. URL можно задавать либо в виде полного адреса Интернета (например, "[http://www.server.ru/imaes/имя\\_файла.gif](http://www.server.ru/imaes/имя_файла.gif)"), либо в виде сокращенного адреса с указанием пути к директории на текущем сервере, в которой хранится данное изображение (например, ".. images/имя\_файла.gif"). Допускается просто указывать имя графического файла, если он хранится в той же директории, что и использующий его файл HTML. Данное изображение может иметь любой размер, поскольку при интерпретации кода оно многократно повторяется, заполняя все доступное пространство в окне броузера. Подробно о правилах включения графических файлов в html-документ мы поговорим в ходе следующего урока.

**СОВЕТ** Для того чтобы избежать неадекватности отображения того или иного цвета различными броузерами, например, когда web-дизайнер решил применить на странице заливку какого-либо редко используемого оттенка, рекомендуется следующий подход: создайте в любом подходящем редакторе графический файл нужного цвета размерами 1x1 пиксел, после чего укажите его в качестве фонового изображения, включив URL этого рисунка в параметр атрибута BACKGROUND тега <BODY>.

Атрибут TEXT позволяет задать цвет текста для всего документа в целом. Но не забывайте, что параметр, назначенный данному атрибуту в теге <BODY>, может быть изменен в определенном участке текста путем использования команды <FONT> с атрибутом COLOR.

Для того чтобы назначить фоновый цвет всему документу, используется атрибут BGCOLOR. В этом случае web-страница будет целиком заполнена равномерной заливкой указанного цвета.

**ПРИМЕЧАНИЕ** Атрибуты BGCOLOR и BACKGROUND не исключают друг друга, однако у последнего имеется приоритет. Это означает, что в случае, когда заданы оба эти атрибута, сначала выполняется заливка web-страницы цветом, назначенным в атрибуте BGCOLOR, поверх которой размещается изображение, заданное атрибутом BACKGROUND.

**ВНИМАНИЕ** Если вы не используете графических изображений в качестве фонового рисунка, а основным цветом html-документа приняли белый, использование атрибута BGCOLOR с параметром "#FFFFFF" в составе тега <BODY> обязательно. Обусловлено это правилом следующей причиной: некоторые броузеры (например, Microsoft Internet Explorer) позволяют пользователям произвольно изменять фоновый цвет загружаемых web-страниц, если он не задан явно. Учтите, что пренебрежение явным указанием фонового цвета может вызвать полное нарушение разработанного вами дизайна.

Атрибут LINK дает web-мастеру возможность назначить цвет, которым отображается не посещенная гиперссылка, то есть ссылка, к которой посетитель данного web-сайта еще не обращался. По умолчанию ей присваивается значение "blue" (0000FF). В свою очередь, атрибут VLINK указывает на цвет посещенной ссылки, значение которой по умолчанию — "purple" (#800080). И наконец, атрибут ALINK определяет цвет активной гиперссылки, то есть цвет, который гиперссылка принимает с момента нажатия на нее курсором мыши до момента загрузки вызываемого ею ресурса. По умолчанию данный атрибут также имеет значение "purple".

Очевидно, что значения всех атрибутов тега <BODY>, кроме атрибута BACKGROUND, представляют собой обозначения цветов символьными метками или шестнадцатеричным цифровым кодом.

**ПРИМЕЧАНИЕ** При выборе цветов документа, текста и гиперссылок следует соблюдать определенную осторожность, поскольку цвета, контрастно отображаемые на цветном мониторе, могут быть неразличимы на черно-белом. Избежать подобных ошибок

можно при помощи несложного приема: сделайте снимок экрана («скриншот») в момент, когда ваша web-страница загружена в браузер (для этого необходимо нажать кнопку Print Screen на клавиатуре компьютера), загрузите полученное изображение в графический редактор и просмотрите его в режиме grayscale (256 оттенков серого). Если выбранная вами цветовая схема по-прежнему выглядит контрастно, ее можно смело применять, если нет — придется использовать другие цвета.

**СОВЕТ** Подбирая цвета для web-страницы, помните, что используемая вами цветовая схема должна быть «корректной». Чтение текста не должно вызывать затруднений, глаза посетителей вашей странички не должны уставать. Пожалеите зрение пользователей, не пишите оранжевым по зеленому.

**ВНИМАНИЕ** Установив одно из значений цветовых параметров вашей страницы, жестко задавайте и остальные значения. Некоторые браузеры позволяют пользователю произвольно менять цвет фона документа или выводимого на экран текста, если они не указаны явно. В этом случае заданный вами цвет текста может совпасть с цветом фона, установленным в браузере пользователя по умолчанию, в результате чего текст станет нечитаемым. Изменение одного из цветовых параметров без изменения остальных недопустимо. Исключение можно сделать лишь в том случае, когда в качестве фонового цвета web-страницы используется белый.

Вот пример использования тега <BODY> со всеми допустимыми атрибутами:

```
<BODY BACKGROUND="http://www.myserver.com/images/back.jpg"
BGCOLOR="#000000" TEXT="#FFFFFF" LINK="#008000" VLINK="#800080"
ALINK="#FF0000">
```

Тэги могут быть пустыми, то есть не содержать текста или других вложенных конструкций (например, <br> который переводит строку). Закрывающий тег в таком случае не указывается.

Также, элементы разметки могут иметь атрибуты, задающие их свойства (например, размер шрифта, цвет, расположение). Атрибуты задаются в начале тега.

<ahref=«http://www.yandex.ru»>Пример элемента с атрибутом href.</a>

Теги можно разделить на следующие группы.

*Гиперссылки*

<AHREF=«filename»target=«\_self»>текст ссылки</A>

где filename – имя файла (может быть и локальным) или адрес страницы в Internet, на который нужно совершить переход.

текст ссылки – текст гипертекстовой ссылки, который будет отображаться в браузере, как правило, выделяется подчеркиванием.

target – задает окно или фрейм, в котором будет открыт документ, при переходе по ссылке. Он может принимать значения:

\_top – документ откроется в текущем окне

\_blank – документ откроется в новом окне

\_self – документ откроется в текущем фрейме

\_parent – документ откроется в родительском фрейме

По умолчанию принимает значение \_self.

*Текстовые ссылки.*

<H1></H1>, <H2></H2>, ... , <H6></H6> – заголовки 1-6 уровней. Применяются для выделения частей выводимого текста (заголовок 1 – будет выведен очень большим, 6 – будет размером сопоставимым с обычным текстом).

**<P>** – обозначает начало нового абзаца. Закрывающий тег **</P>**, не является обязательным.

**<BR>** – переход на новую строку. Закрывающий тег **</br>**, отсутствует.

**<HR>** – горизонтальная линия.

**<BLOCKQUOTE></BLOCKQUOTE>** – цитата. Выделение заданного текста как цитаты.

**<PRE></PRE>** – режим предпросмотра. При этом текст заключается в рамку и выводится не форматированным (то есть все теги, кроме **</PRE>**, игнорируются, но при этом переводы строки ставятся там, где они присутствуют в исходном документе).

**<DIV></DIV>** – блок текста (как правило, применяется для использования каскадных стилей CSS).

**<SPAN></SPAN>** – строка (как правило, применяется для использования каскадных стилей CSS).

*Теги форматирования текста*

**<EM></EM>** – выделение символа, на который падает ударение (обычно отображается курсивом).

**<STRONG></STRONG>** – выделение символа, на который падает усиленное ударение (обычно отображается жирным текстом).

**<I></I>** – выделение текста курсивом.

**<B></B>** – выделение текста жирным шрифтом.

**<U></U>** – подчёркивание текста

**<S></S>** – зачёркивание текста.

**<STRIKE></STRIKE>** – то же самое, что **<S> ... </S>**

**<BIG></BIG>** – увеличение шрифта.

**<SMALL></SMALL>** – уменьшение шрифта.

**<BLINK></BLINK>** – мигающий текст.

**<MARQUEE></MARQUEE>** – сдвигающийся по экрану текст.

**<SUB></SUB>** – вывод текста под строкой. Например, **H<SUB>2</SUB>** Отобразится в виде текста H<sub>2</sub>O.

**<SUP></SUP>** – вывод текста над строкой. Например, **E=mc<SUP>2</SUP>** отобразится в виде текста E=mc<sup>2</sup>.

**<FONT атрибуты></FONT>** – задание атрибутов у используемого шрифта. Атрибуты могут быть следующими:

**COLOR=color** – указание цвета. Цвет может быть указан шестнадцатеричным числом в формате #rrggbb (первые 2 шестнадцатеричные цифры указывают интенсивность красного, следующие 2 – зелёного, последние 2 – синего) или названием самого цвета.

**FACE=** указываем имя шрифта.

**SIZE=** позволяет изменить размер шрифта. Размеры могут быть от 1 до 7, по умолчанию размер 3.

**SIZE=+ размер** или **SIZE=-размер** – размер больше или меньше стандартного. Например, **SIZE=+2** указывает размер на 2 больше стандарта, то есть размер 5.

### Списки.

Данная конструкция

**<UL>**

**<LI>** первый элемент списка **</LI>**

**<LI>** второй элемент списка **</LI>**

**<LI>** третий элемент списка **</LI>**

**</UL>**

создаёт список вида:

- первый элемент;

- второй элемент;

- третий элемент.

Также стоит отметить, что тегов есть параметры, позволяющие менять вид списка.

*Объекты.*

EMBED – вставка объектов различных типов

APPLET – вставка Java-апплетов

SCRIPT – вставка различных скриптов, например JavaScript

*Изображения.*

IMG – тег для вставки изображения. Это не закрывающийся тег.

SRC – имя локального файла или путь к нему в виде URL

ALT – текст картинки (отобразится, в виде текста, если не удалось отобразить картинку)

TITLE – подсказка (показывается при попадании курсора в область картинки)

WIDTH, HEIGHT – размеры изображения (выводимое изображение будет масштабировано до указанных размеров)

ALIGN – обтекание текста

*Таблицы.*

TABLE – тег создание таблицы. Тег имеет следующие параметры:

BORDER – задает толщину границу таблицы

CELLSPACING – задает расстояние от ячейки до ячейки

CAPTION – задает заголовок таблицы (необязательный тег)

TR – добавление строки в таблицы

TH – задает заголовок столбца (необязательный тег)

TD – добавление ячейки таблицы

WIDTH, HEIGHT – размеры таблицы

*Формы.* Формы ввода данных могут быть самыми разнообразными. Поэтому рассмотрим только основные теги:

FORM – тег для создания формы

INPUT – добавление элемента ввода

TEXTAREA – добавление текстового поля

SELECT – добавление списка (как правило, это выпадающее меню)

OPTION – пункт списка

*Символы.* Некоторые символы не могут быть выведены напрямую. Для их вывода требуется использовать их определения, например, символ амперсанд & в коде HTML будет иметь вид **&amp;**, символ меньше < будет иметь вид **&lt;**, символ больше > будет **&gt;**. Это ограничение введено, так как эти символы уже используются в языке HTML как служебные.

Любая HTML-страница должна иметь обозначение начала и конца документа обрамленные тегами **<html>** и **</html>** соответственно. Внутри них должны находиться теги заголовка **<head>** и **</head>**, и теги, обозначающие тело документа **<body>** и **</body>**. А внутри них могут быть произвольные комбинации из групп тегов описанных ранее.

Также рассмотрим ExtensibleHypertextMarkupLanguage (XHTML) это расширяемый язык разметки гипертекста. Стоит отметить, что язык XHTML это не описание самого языка, а список отличий XHTML от HTML. Рассмотрим основные отличия HTML и XHTML. В XHTML все используемые теги должны иметь закрывающий тег. Теги, не имеющие закрывающего тега должны оканчиваться символом /. Например тег **<br>**, должен иметь закрывающий его тег **<br />**. В XHTML допускается писать теги и их атрибуты только строчными буквами. В XHTML очень строгая проверка синтаксиса не допускается использовать < и &, даже в URL, вместо них должны быть **&lt;** и **&amp;**. Браузеры, обнаружив ошибку синтаксиса в XHTML, должны прекратить его обработку и вывести ошибку на экран. В стандарте HTML браузер должен попытаться отобразить

запрашиваемый документ. Стоит отметить, что XHTML расширяемый язык – за счет указания типа документа и возможности использовать свои теги.

Как мы видим, язык разметки HTML предоставляет широкие возможности для отображения информации, для этого в нем содержится большое количество тегов для различного форматирования выводимой информации. Язык XHTML очень похож на HTML, но более строгий, грамматические правила в XHTML менее сложные, и как следствие при создании Web-страниц будет меньше ошибок.

### **Ход работы:**

#### **I. Создание простейших файлов HTML.**

1. Создаём папку, в которой будем сохранять созданные Web-страницы.
2. Запускаем программу Notepad++ и набираем следующий текст с элементами форматирования:

```
<HTML>
<HEAD>
<TITLE> Учебный файл HTML </TITLE>
</HEAD>
<BODY>
Расписание занятий на вторник
</BODY>
</HTML>
```

3. Сохраняем файл под именем *schedule.html*.

4. Для просмотра созданной Web-страницы используем браузер (Рис.1).

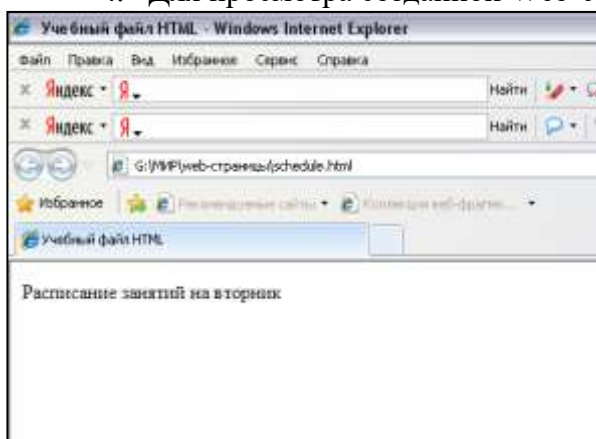


Рис.1 «Учебный файл HTML»

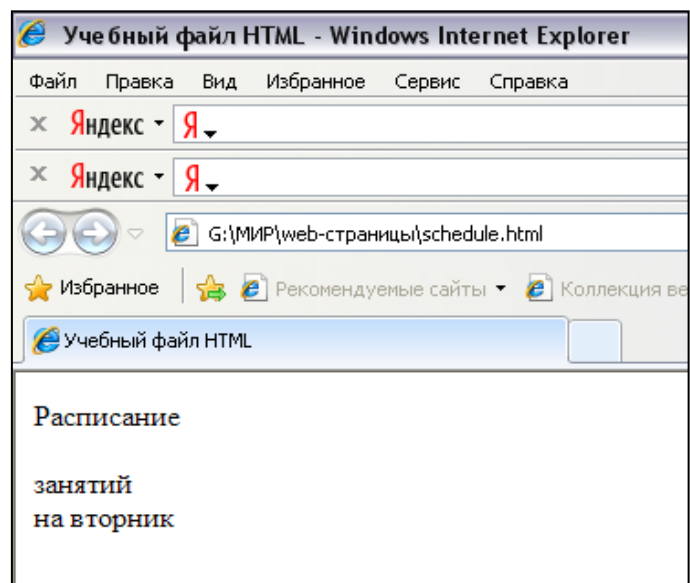
#### **II. Управление расположением текста на экране.**

1. Вносим изменения в текст, расположив слова "Расписание", "занятий", "на вторник" на разных строках:

```
<HTML>
<HEAD>
<TITLE> Учебный файл HTML </TITLE>
</HEAD>
<BODY>
Расписание
занятий
на вторник
</BODY>
</HTML>
```

2. Сохраняем внесенные изменения.
3. Новая полученная Web-страница не изменилась (см. Рис.1). Для того, чтобы переносить текст используют теги. Тег перевода строки <BR> отделяет строку от последующего текста или графики. Тег абзаца <P> тоже отделяет строку, но еще добавляет пустую строку..
4. Вносим изменения в текст файла HTML:  
<HTML>  
<HEAD>  
<TITLE> Учебный файл HTML </TITLE>  
</HEAD>  
<BODY>  
Расписание <P>занятий <BR>на вторник  
</BODY>  
</HTML>

Рис.2 «Изменённый файл»



### III. Выделение фрагментов текста.

Существует три тега выделения фрагментов текста: **<B> : </B>** — для выделения полужирным, **<I> : </I>** — для выделения курсивом, **<U> : </U>** — для выделения подчеркиванием.

1. Вносим изменения в файл *schedule.html*: Рис.3

```
<HTML>
<HEAD>
<TITLE> Учебный файл HTML </TITLE>
</HEAD>
<BODY>
<B>Расписание</B> <I>занятий</I> <U> на вторник</U>
</BODY>
</HTML>
```

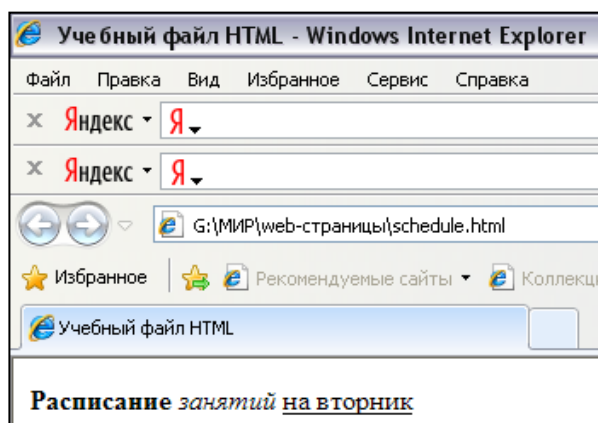


Рис. 3 «Новый вид»

2. Также используем вложение тегов:

**<I><B>Расписание</B></I> <I>занятий</I> <U> на вторник</U>** (Рис.4).

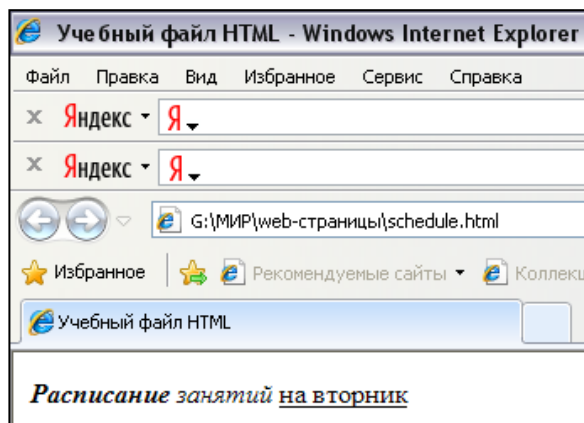


Рис.4 «Ещё один вариант»

## Базовый синтаксис CSS

Как уже было отмечено ранее, стилевые правила записываются в своём формате, отличном от HTML. Основным понятием выступает селектор — это некоторое имя стиля, для которого добавляются параметры форматирования. В качестве селектора выступают теги, классы и идентификаторы. Общий способ записи имеет следующий вид.

селектор                      свойство                      значение

body { background: #ffc910; }

Вначале пишется имя селектора, например, TABLE, это означает, что все стилевые параметры будут применяться к тегу <table>, затем идут фигурные скобки, в которых записывается стилевое свойство, а его значение указывается после двоеточия. Стилиевые свойства разделяются между собой точкой с запятой, в конце этот символ можно опустить.

CSS не чувствителен к регистру, переносу строк, пробелам и символам табуляции, поэтому форма записи зависит от желания разработчика. Так, в примере 5.1 показаны две разновидности оформления селекторов и их правил.

### Пример 5.1. Использование стилей

HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Заголовки</title>
    <style>
      h1 { color: #a6780a; font-weight: normal; }
      h2 {
        color: olive;
        border-bottom: 2px solid black;
      }
    </style>
  </head>
  <body>

    <h1>Заголовок 1</h1>
    <h2>Заголовок 2</h2>

  </body>
</html>
```

В данном примере свойства селектора h1 записаны в одну строку, а для селектора h2 каждое свойство находится на отдельной строке. Во втором случае легче отыскивать нужные свойства и править их по необходимости, но при этом незначительно возрастает объем данных за счёт активного использования пробелов и переносов строк. Так что в любом случае способ оформления стилиевых параметров зависит от разработчика.

## Правила применения стилей

Далее приведены некоторые правила, которые необходимо знать при описании стиля.

### Форма записи

Для селектора допускается добавлять каждое стилевое свойство и его значение по отдельности, как это показано в примере 5.2.

### Пример 5.2. Расширенная форма записи

```
td { background: olive; }
td { color: white; }
td { border: 1px solid black; }
```

Однако такая запись не очень удобна. Приходится повторять несколько раз один и тот же селектор, да и легко запутаться в их количестве. Поэтому пишите все свойства для каждого селектора вместе. Указанный набор записей в таком случае получит следующий вид (пример 5.3).

### Пример 5.3. Компактная форма записи

```
td {
  background: olive;
  color: white;
  border: 1px solid black;
}
```

Эта форма записи более наглядная и удобная в использовании.

### Имеет приоритет значение, указанное в коде ниже

Если для селектора вначале задаётся свойство с одним значением, а затем то же свойство, но уже с другим значением, то применяться будет то значение, которое в коде установлено ниже (пример 5.4).

### Пример 5.4. Разные значения у одного свойства

```
p { color: green; }
p { color: red; }
```

В данном примере для селектора `p` цвет текста вначале установлен зелёным, а затем красным. Поскольку значение `red` расположено ниже, то оно в итоге и будет применяться к тексту.

На самом деле такой записи лучше вообще избегать и удалять повторяющиеся значения. Но подобное может произойти случайно, например, в случае подключения разных стилевых файлов, в которых содержатся одинаковые селекторы.

### Значения

У каждого свойства может быть только соответствующее его функции значение. Например, для `color`, который устанавливает цвет текста, в качестве значений недопустимо использовать числа.

## Комментарии

Комментарии нужны, чтобы делать пояснения по поводу использования того или иного стилевого свойства, выделять разделы или писать свои заметки. Комментарии позволяют легко вспоминать логику и структуру селекторов, и повышают разборчивость кода. Вместе с тем, добавление текста увеличивает объём документов, что отрицательно сказывается на времени их загрузки. Поэтому комментарии обычно применяют в отладочных или учебных целях, а при выкладывании сайта в сеть их стирают.

Чтобы пометить, что текст является комментарием, применяют следующую конструкцию `/* ... */` (пример 5.5).

#### Пример 5.5. Комментарии в CSS-файле

```
/*
  Стил ь для сайта htmlbook.ru
  Сделан для ознакомительных целей
*/

div {
  width: 200px; /* Ширина блока */
  margin: 10px; /* Поля вокруг элемента */
  float: left; /* Обтекание по правому краю */
}
```

Как следует из данного примера, комментарии можно добавлять в любое место CSS-документа, а также писать текст комментария в несколько строк. Вложенные комментарии недопустимы.

## Значения стилевых свойств

Всё многообразие значений стилевых свойств может быть сведено к определённому типу: строка, число, проценты, размер, цвет, адрес или ключевое слово.

## Строки

Любые строки необходимо брать в двойные или одинарные кавычки. Если внутри строки требуется оставить одну или несколько кавычек, то можно комбинировать типы кавычек или добавить перед кавычкой слэш (пример 6.1).

#### Пример 6.1. Допустимые строки

```
'Гостиница "Турист"'
"Гостиница 'Турист'"
"Гостиница \"Турист\""
```

В данном примере в первой строке применяются одинарные кавычки, а слово «Турист» взято в двойные кавычки. Во второй строке всё с точностью до наоборот, в третьей же строке используются только двойные кавычки, но внутренние экранированы с помощью слэша.

## Числа

Значением может выступать целое число, содержащее цифры от 0 до 9 и десятичная дробь, в которой целая и десятичная часть разделяются точкой (пример 6.2).

## Пример 6.2. Числа в качестве значений

### HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Числа</title>
    <style>
      p {
        font-weight: 600; /* Жирное начертание */
        line-height: 1.2; /* Межстрочный интервал */
      }
    </style>
  </head>
  <body>
    <p>Пример текста</p>
  </body>
</html>
```

Если в десятичной дроби целая часть равна нулю, то её разрешается не писать. Запись .7 и 0.7 равнозначна.

## Проценты

Процентная запись обычно применяется в тех случаях, когда надо изменить значение относительно родительского элемента или когда размеры зависят от внешних условий. Так, ширина таблицы 100% означает, что она будет подстраиваться под размеры окна браузера и меняться вместе с шириной окна (пример 6.3).

## Пример 6.3. Процентная запись

### HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ширина в процентах</title>
    <style>
      TABLE {
        width: 100%; /* Ширина таблицы в процентах */
        background: #f0f0f0; /* Цвет фона */
      }
    </style>
  </head>
  <body>
    <table>
      <tr><td>Содержимое таблицы</td></tr>
    </table>
  </body>
</html>
```

Проценты не обязательно должны быть целым числом, допускается использовать десятичные дроби, вроде значения 56.8%.

# Размеры

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

Относительные единицы обычно используют для работы с текстом, либо когда надо вычислить процентное соотношение между элементами. В табл. 6.1 перечислены основные относительные единицы.

Табл. 6.1. Относительные единицы измерения

| Единица | Описание                        |
|---------|---------------------------------|
| em      | Размер шрифта текущего элемента |
| ex      | Высота символа x                |
| px      | Пиксел                          |
| %       | Процент                         |

Единица em это изменяемое значение, которое зависит от размера шрифта текущего элемента (размер устанавливается через стилевое свойство font-size). 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 или размеру шрифта родительского элемента. Процентная запись идентична em, в том смысле, что значения 1em и 100% равны.

Единица 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.

Пиксел это элементарная точка, отображаемая монитором или другим подобным устройством, например, смартфоном. Размер пиксела зависит от разрешения устройства и его технических характеристик. В примере 6.4 показано применение пикселей и em для задания размера шрифта.

### Пример 6.4. Использование относительных единиц

HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
<head>
  <meta charset="utf-8">
  <title>Относительные единицы</title>
  <style>
    H1 { font-size: 30px; }
    P { font-size: 1.5em; }
  </style>
</head>
<body>
  <h1>Заголовок размером 30 пикселей</h1>
```

```

    <p>Размер текста 1.5 em</p>
  </body>
</html>

```

Результат данного примера показан ниже (рис. 6.1).

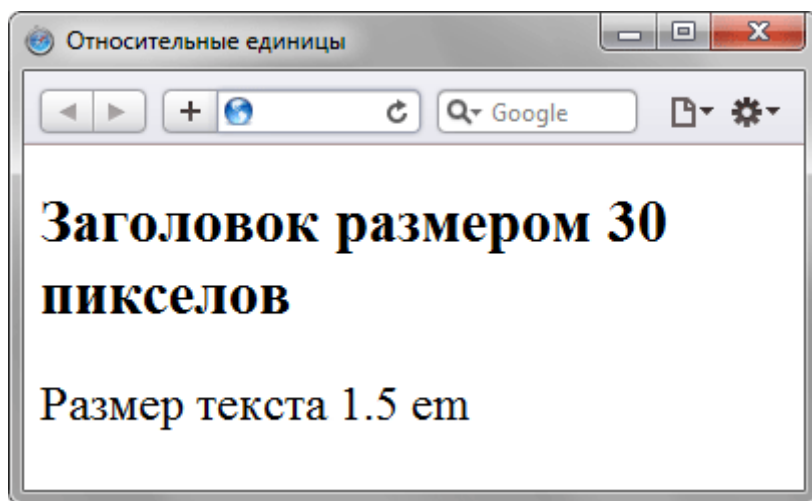


Рис. 6.1. Размер текста при различных единицах

## Абсолютные единицы

Абсолютные единицы применяются реже, чем относительные и обычно при работе с текстом. В табл. 6.2 перечислены основные абсолютные единицы.

Табл. 6.2. Абсолютные единицы измерения

| 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)   |

Самой, пожалуй, распространенной единицей является пункт, который используется для указания размера шрифта. Хотя мы привыкли измерять все в миллиметрах и подобных единицах, пункт, пожалуй, единственная величина из не метрической системы измерения, которая используется у нас повсеместно. И все благодаря текстовым редакторам и издательским системам. В примере 6.5 показано использование пунктов и миллиметров.

Пример 6.5. Использование абсолютных единиц

## HTML5CSS 2.1IECrOpSaFx

```

<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Абсолютные единицы</title>
    <style>
      h1 { font-size: 24pt; }
    </style>
  </head>
  <body>

```

```

P { margin-left: 30mm; }
</style>
</head>
<body>
  <h1>Заголовок размером 24 пункта</h1>
  <p>Сдвиг текста вправо на 30 миллиметров</p>
</body>
</html>

```

Результат использования абсолютных единиц измерения показан ниже (рис. 6.2).

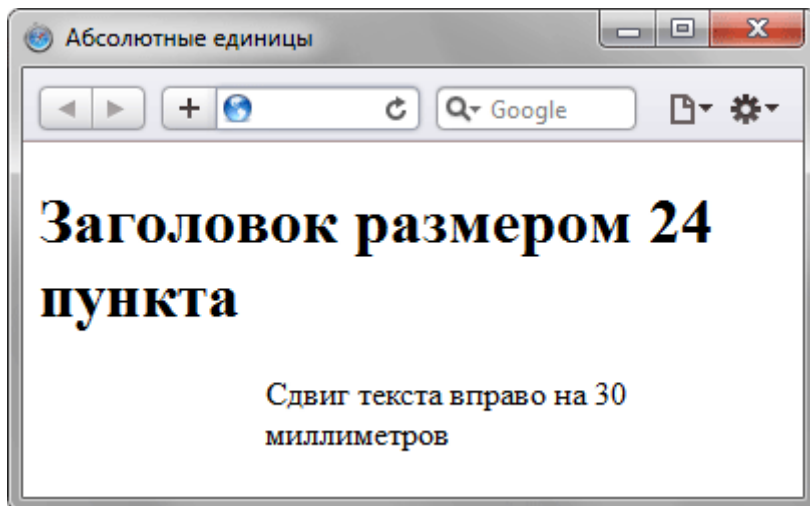


Рис. 6.2. Размер текста при различных единицах

При установке размеров обязательно указывайте единицы измерения, например `width: 30px`. В противном случае браузер не сможет показать желаемый результат, поскольку не понимает, какой размер вам требуется. Единицы не добавляются только при нулевом значении (`margin: 0`).

## Цвет

Цвет в стилях можно задавать тремя способами: по шестнадцатеричному значению, по названию и в формате RGB.

### По шестнадцатеричному значению

Для задания цветов используются числа в шестнадцатеричном коде. Шестнадцатеричная система, в отличие от десятичной системы, базируется, как следует из ее названия, на числе 16. Цифры будут следующие: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Цифры от 10 до 15 заменены латинскими буквами. Числа больше 15 в шестнадцатеричной системе образуются объединением двух чисел в одно. Например, числу 255 в десятичной системе соответствует число FF в шестнадцатеричной системе. Чтобы не возникало путаницы в определении системы счисления, перед шестнадцатеричным числом ставят символ решетки #, например #666999. Каждый из трех цветов — красный, зеленый и синий — может принимать значения от 00 до FF. Таким образом, обозначение цвета разбивается на три составляющие #rrggbb, где первые два символа отмечают красную компоненту цвета, два средних — зелёную, а два последних — синюю. Допускается использовать сокращенную форму вида #rgb, где каждый символ следует удваивать (#rrggbb). К примеру, запись #fe0 расценивается как #ffee00.

## По названию

Браузеры поддерживают некоторые цвета по их названию. В табл. 6.3 приведены названия, шестнадцатеричный код и описание.

Табл. 6.3. Названия цветов

| Имя     | Цвет                                                                                | Код              | Описание       |
|---------|-------------------------------------------------------------------------------------|------------------|----------------|
| white   |                                                                                     | #ffffff или #fff | Белый          |
| silver  |    | #c0c0c0          | Серый          |
| gray    |    | #808080          | Тёмно-серый    |
| black   |    | #000000 или #000 | Чёрный         |
| maroon  |    | #800000          | Тёмно-красный  |
| red     |    | #ff0000 или #f00 | Красный        |
| orange  |    | #ffa500          | Оранжевый      |
| yellow  |    | #ffff00 или #ff0 | Жёлтый         |
| olive   |    | #808000          | Оливковый      |
| lime    |    | #00ff00 или #0f0 | Светло-зелёный |
| green   |    | #008000          | Зелёный        |
| aqua    |    | #00ffff или #0ff | Голубой        |
| blue    |   | #0000ff или #00f | Синий          |
| navy    |  | #000080          | Тёмно-синий    |
| teal    |  | #008080          | Сине-зелёный   |
| fuchsia |  | #ff00ff или #f0f | Розовый        |
| purple  |  | #800080          | Фиолетовый     |

## С помощью RGB

Можно определить цвет, используя значения красной, зелёной и синей составляющей в десятичном исчислении. Значение каждого из трех цветов может принимать значения от 0 до 255. Также можно задавать цвет в процентном отношении. Вначале указывается ключевое слово `rgb`, а затем в скобках, через запятую указываются компоненты цвета, например `rgb(255, 0, 0)` или `rgb(100%, 20%, 20%)`.

В примере 6.6 представлены различные способы задания цветов элементов веб-страниц.

Пример 6.6. Представление цвета

HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
<head>
  <meta charset="utf-8">
  <title>Цвета</title>
  <style>
    BODY {
      background-color: #3366CC; /* Цвет фона веб-страницы */
    }
    H1 {
      background-color: RGB(249, 201, 16); /* Цвет фона под заголовком */
    }
  </style>
</head>
<body>
  <h1>Пример 6.6. Представление цвета</h1>
</body>
</html>
```

```

    }
    P {
        background-color: maroon; /* Цвет фона под текстом абзаца */
        color: white; /* Цвет текста */
    }
</style>
</head>
<body>
    <h1>Lorem ipsum dolor sit amet</h1>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam
        nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
        volutpat.</p>
</body>
</html>

```

Результат данного примера показан на рис. 6.3.



Рис. 6.3. Цвета на веб-странице

## Адреса

Адреса (URI, Uniform Resource Identifiers, унифицированный идентификатор ресурсов) применяются для указания пути к файлу, например, для установки фоновой картинки на странице. Для этого применяется ключевое слово `url()`, внутри скобок пишется относительный или абсолютный адрес файла. При этом адрес можно задавать в необязательных одинарных или двойных кавычках (пример 6.7).

Пример 6.7. Адрес графического файла

HTML5CSS 2.1IECrOpSaFx

```

<!DOCTYPE HTML>
<html>
<head>
    <meta charset="utf-8">
    <title>Добавление фона</title>
    <style>
        body {
            background: url('http://webimg.ru/images/156_1.png') no-repeat;
        }
    
```

```

div {
  background: url(images/warning.png) no-repeat;
  padding-left: 20px;
  margin-left: 200px;
}
</style>
</head>
<body>
  <div>Внимание, запрашиваемая страница не найдена!</div>
</body>
</html>

```

В данном примере в селекторе `body` используется абсолютный адрес к графическому файлу, а в селекторе `div` — относительный.

## Ключевые слова

В качестве значений активно применяются ключевые слова, которые определяют желаемый результат действия стиливых свойств. Ключевые слова пишутся без кавычек.

Правильно: `P { text-align: right; }`

Неверно: `P { text-align: "right"; }`

## Селекторы тегов

В качестве селектора может выступать любой тег HTML, для которого определяются правила форматирования, такие как: цвет, фон, размер и т. д. Правила задаются в следующем виде.

Тег { свойство1: значение; свойство2: значение; ... }

Вначале указывается имя тега, оформление которого будет переопределено, заглавными или строчными символами не имеет значения. Внутри фигурных скобок пишется стиливое свойство, а после двоеточия — его значение. Набор свойств разделяется между собой точкой с запятой и может располагаться как в одну строку, так и в несколько (пример 7.1).

Пример 7.1. Изменение стиля тега абзаца

HTML5CSS 2.1IECrOpSaFx

```

<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Селекторы тегов</title>
    <style>
      P {
        text-align: justify; /* Выравнивание по ширине */
        color: green; /* Зеленый цвет текста */
      }
    </style>
  </head>

```

```
<body>

  <p>Более эффективным способом ловли льва в пустыне
  является метод золотого сечения. При его использовании пустыня делится
  на две неравные части, размер которых подчиняется правилу золотого
  сечения.</p>

</body>
</html>
```

В данном примере изменяется цвет и выравнивание текста абзаца. Стилль будет применяться только к тексту, который располагается внутри контейнера `<p>`.

Следует понимать, что хотя стиль можно применить к любому тегу, результат будет заметен только для тегов, которые непосредственно отображаются в контейнере `<body>`.

## Классы

Классы применяют, когда необходимо определить стиль для индивидуального элемента веб-страницы или задать разные стили для одного тега. При использовании совместно с тегами синтаксис для классов будет следующий.

Тег.Имя класса { свойство1: значение; свойство2: значение; ... }

Внутри стиля вначале пишется желаемый тег, а затем, через точку пользовательское имя класса. Имена классов должны начинаться с латинского символа и могут содержать в себе символ дефиса (-) и подчеркивания (\_). Использование русских букв в именах классов недопустимо. Чтобы указать в коде HTML, что тег используется с определённым классом, к тегу добавляется атрибут `class="Имя класса"` (пример 8.1).

### Пример 8.1. Использование классов

#### HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Классы</title>
    <style>
      P { /* Обычный абзац */
        text-align: justify; /* Выравнивание текста по ширине */
      }
      P.cite { /* Абзац с классом cite */
        color: navy; /* Цвет текста */
        margin-left: 20px; /* Отступ слева */
        border-left: 1px solid navy; /* Граница слева от текста */
        padding-left: 15px; /* Расстояние от линии до текста */
      }
    </style>
  </head>
  <body>
    <p>Более эффективным способом ловли льва в пустыне
    является метод золотого сечения. При его использовании пустыня делится
    на две неравные части, размер которых подчиняется правилу золотого
    сечения.</p>
    <p class="cite">Более эффективным способом ловли льва в пустыне
    является метод золотого сечения. При его использовании пустыня делится
    на две неравные части, размер которых подчиняется правилу золотого
    сечения.</p>
  </body>
</html>
```

```

</style>
</head>
<body>
  <p>Для искусственного освещения помещения применяются люминесцентные лампы.
    Они отличаются высокой световой отдачей, продолжительным сроком службы,
    малой яркостью светящейся поверхности, близким к естественному
    спектральным
    составом излучаемого света, что обеспечивает хорошую цветопередачу.</p>
  <p class="cite">Для исключения засветки экрана дисплея световыми потоками
    оконные проемы снабжены светорассеивающими шторами.</p>
</body>
</html>

```

Результат данного примера показан на рис. 8.1.

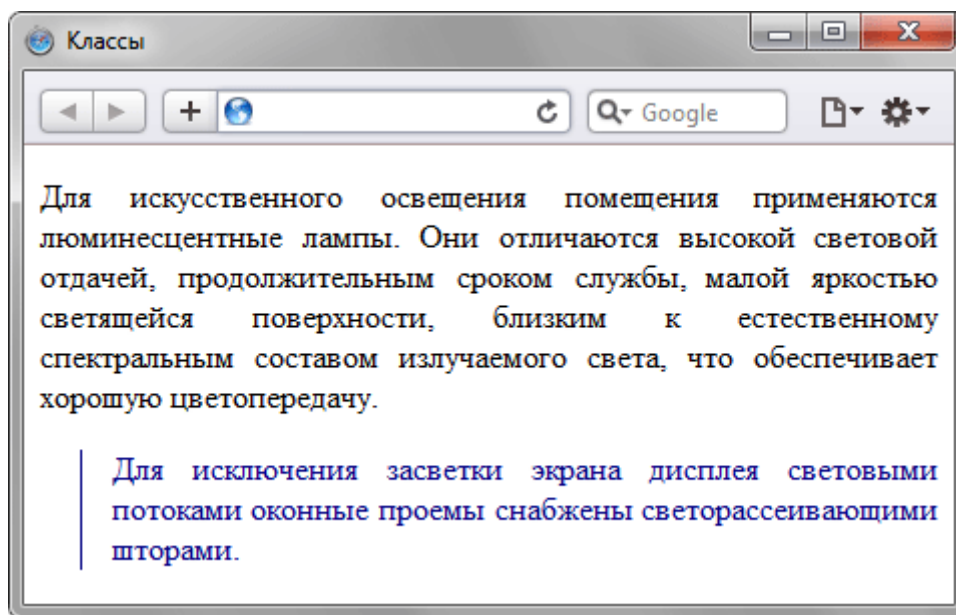


Рис. 8.1. Вид текста, оформленного с помощью стилевых классов

Первый абзац выровнен по ширине с текстом чёрного цвета (этот цвет задаётся браузером по умолчанию), а следующий, к которому применен класс с именем `cite` — отображается синим цветом и с линией слева.

Можно, также, использовать классы и без указания тега. Синтаксис в этом случае будет следующий.

.Имя класса { свойство1: значение; свойство2: значение; ... }

При такой записи класс можно применять к любому тегу (пример 8.2).

Пример 8.2. Использование классов

HTML5CSS 2.1IECrOpSaFx

```

<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Классы</title>
    <style>

```

```

.gost {
  color: green; /* Цвет текста */
  font-weight: bold; /* Жирное начертание */
}
.term {
  border-bottom: 1px dashed red; /* Подчеркивание под текстом */
}
</style>
</head>
<body>
  <p>Согласно <span class="gost">ГОСТ 12.1.003-83 ССБТ &quot;Шум. Общие
    требования безопасности&quot;</span>, шумовой характеристикой рабочих
    мест при постоянном шуме являются уровни звуковых давлений в децибелах
    в октавных полосах. Совокупность таких уровней называется
    <b class="term">предельным спектром</b>, номер которого численно равен
    уровню звукового давления в октавной полосе со среднегеометрической
    частотой 1000&nbsp;Гц.
  </p>
</body>
</html>

```

Результат применения классов к тегам `<span>` и `<b>` показан на рис. 8.2.

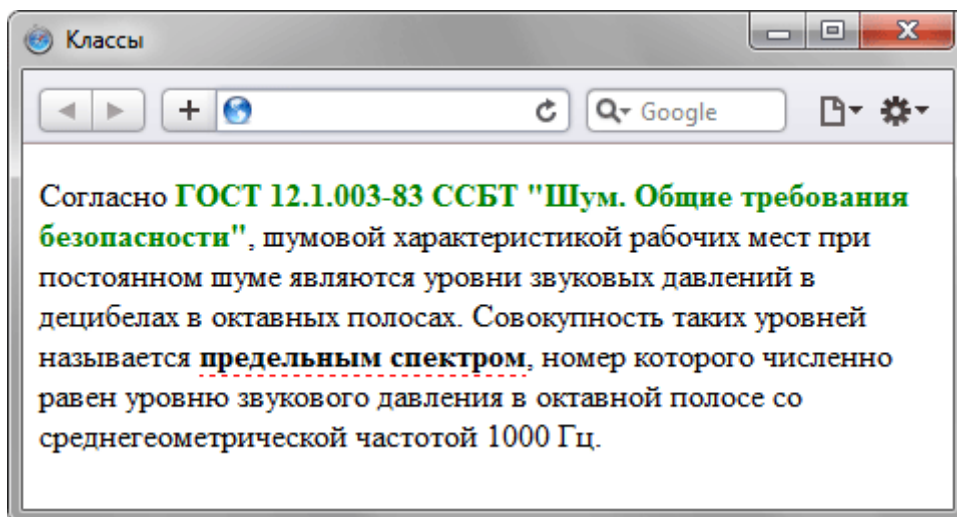


Рис. 8.2. Вид тегов, оформленных с помощью классов

Классы удобно использовать, когда нужно применить стиль к разным элементам веб-страницы: ячейкам таблицы, ссылкам, абзацам и др. В примере 8.3 показано изменение цвета фона строк таблицы для создания «зебры».

### Пример 8.3. Использование классов

#### HTML5CSS 2.1IECrOpSaFx

```

<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Камни</title>
    <style>
      table.jewel {
        width: 100%; /* Ширина таблицы */
        border: 1px solid #666; /* Рамка вокруг таблицы */
      }
    </style>
  </head>
  <body>
    <table class="jewel">
      <tr>
        <td>Камни</td>
      </tr>
    </table>
  </body>
</html>

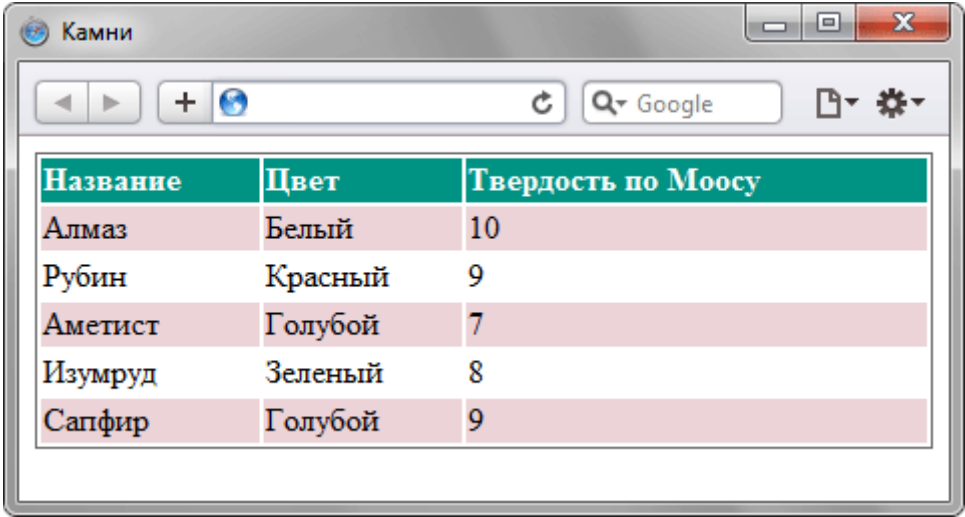
```

```

th {
  background: #009383; /* Цвет фона */
  color: #fff; /* Цвет текста */
  text-align: left; /* Выравнивание по левому краю */
}
tr.odd {
  background: #ebd3d7; /* Цвет фона */
}
</style>
</head>
<body>
<table class="jewel">
  <tr>
    <th>Название</th><th>Цвет</th><th>Твердость по Моосу</th>
  </tr>
  <tr class="odd">
    <td>Алмаз</td><td>Белый</td><td>10</td>
  </tr>
  <tr>
    <td>Рубин</td><td>Красный</td><td>9</td>
  </tr>
  <tr class="odd">
    <td>Аметист</td><td>Голубой</td><td>7</td>
  </tr>
  <tr>
    <td>Изумруд</td><td>Зеленый</td><td>8</td>
  </tr>
  <tr class="odd">
    <td>Сапфир</td><td>Голубой</td><td>9</td>
  </tr>
</table>
</body>
</html>

```

Результат данного примера показан на рис. 8.3. В примере класс с именем odd используется для изменения цвета фона строки таблицы. За счёт того, что этот класс добавляется не ко всем тегам <tr> и получается чередование разных цветов.



| Название | Цвет    | Твердость по Моосу |
|----------|---------|--------------------|
| Алмаз    | Белый   | 10                 |
| Рубин    | Красный | 9                  |
| Аметист  | Голубой | 7                  |
| Изумруд  | Зеленый | 8                  |
| Сапфир   | Голубой | 9                  |

Рис. 8.3. Результат применения классов

## Одновременное использование разных классов

К любому тегу одновременно можно добавить несколько классов, перечисляя их в атрибуте class через пробел. В этом случае к элементу применяется стиль, описанный в правилах для каждого класса. Поскольку при добавлении нескольких классов они могут содержать одинаковые стилевые свойства, но с разными значениями, то берётся значение у класса, который описан в коде ниже.

В примере 8.4 показано использование разных классов для создания облака тегов.

#### Пример 8.4. Сочетание разных классов

##### HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
<head>
  <meta charset="utf-8">
  <title>Облако тегов</title>
  <style type="text/css">
    .level1 { font-size: 1em; }
    .level2 { font-size: 1.2em; }
    .level3 { font-size: 1.4em; }
    .level4 { font-size: 1.6em; }
    .level5 { font-size: 1.8em; }
    .level6 { font-size: 2em; }
    A.tag {
      color: #468be1; /* Цвет ссылок */
    }
  </style>
</head>
<body>
<div>
  <a href="/term/2" class="tag level6">Paint.NET</a>
  <a href="/term/69" class="tag level6">Photoshop</a>
  <a href="/term/3" class="tag level5">цвет</a>
  <a href="/term/95" class="tag level5">фон</a>
  <a href="/term/11" class="tag level4">палитра</a>
  <a href="/term/43" class="tag level3">слои</a>
  <a href="/term/97" class="tag level2">свет</a>
  <a href="/term/44" class="tag level2">панели</a>
  <a href="/term/16" class="tag level1">линия</a>
  <a href="/term/33" class="tag level1">прямоугольник</a>
  <a href="/term/14" class="tag level1">пиксел</a>
  <a href="/term/27" class="tag level1">градиент</a>
</div>
</body>
</html>
```

Результат данного примера показан на рис. 8.4.

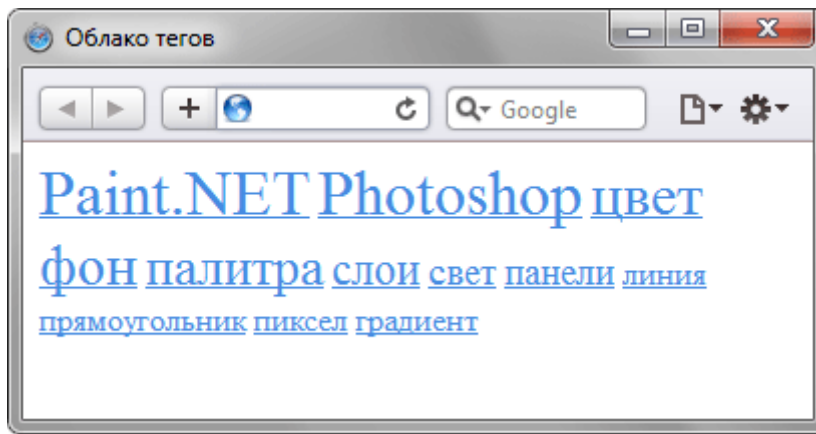


Рис. 8.4. Облако тегов

В стилях также допускается использовать запись вида `.layer1.layer2`, где `layer1` и `layer2` представляют собой имена классов. Стиль применяется только для элементов, у которых одновременно заданы классы `layer1` и `layer2`.

## Идентификаторы

Идентификатор (называемый также «ID селектор») определяет уникальное имя элемента, которое используется для изменения его стиля и обращения к нему через скрипты.

Синтаксис применения идентификатора следующий.

`#Имя идентификатора { свойство1: значение; свойство2: значение; ... }`

При описании идентификатора вначале указывается символ решётки (`#`), затем идет имя идентификатора. Оно должно начинаться с латинского символа и может содержать в себе символ дефиса (`-`) и подчеркивания (`_`). Использование русских букв в именах идентификатора недопустимо. В отличие от классов идентификаторы должны быть уникальны, иными словами, встречаться в коде документа только один раз.

Обращение к идентификатору происходит аналогично классам, но в качестве ключевого слова у тега используется атрибут `id`, значением которого выступает имя идентификатора (пример 9.1). Символ решётки при этом уже не указывается.

Пример 9.1. Использование идентификатора

HTML5CSS 2.1IECrOpSaFx

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Идентификаторы</title>
    <style>
```

```

#help {
  position: absolute; /* Абсолютное позиционирование */
  left: 160px; /* Положение элемента от левого края */
  top: 50px; /* Положение от верхнего края */
  width: 225px; /* Ширина блока */
  padding: 5px; /* Поля вокруг текста */
  background: #f0f0f0; /* Цвет фона */
}
</style>
</head>
<body>
<div id="help">
  Этот элемент помогает в случае, когда вы находитесь в осознании того
  факта, что совершенно не понимаете, кто и как вам может помочь. Именно
  в этот момент мы и подсказываем, что помочь вам никто не сможет.
</div>

</body>
</html>

```

В данном примере определяется стиль тега `<div>` через идентификатор с именем `help` (рис. 9.1).

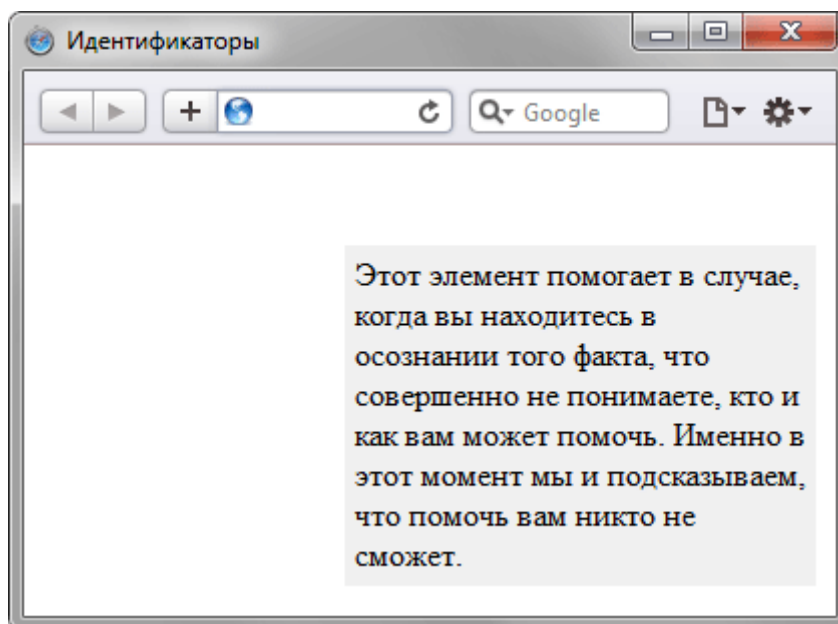


Рис. 9.1. Результат применения идентификатора

Как и при использовании классов, идентификаторы можно применять к конкретному тегу. Синтаксис при этом будет следующий.

Тег#Имя идентификатора { свойство1: значение; свойство2: значение; ... }

Вначале указывается имя тега, затем без пробелов символ решётки и название идентификатора. В примере 9.2 показано использование идентификатора применительно к тегу `<p>`.

Пример 9.2. Идентификатор совместно с тегом

HTML5CSS 2.1IECrOpSaFx

```

<!DOCTYPE HTML>
<html>
<head>
  <meta charset="utf-8">
  <title>Идентификаторы</title>
  <style>
    P {
      color: green; /* Зеленый цвет текста */
      font-style: italic; /* Курсивное начертание текста */
    }
    P#ора {
      color: red; /* Красный цвет текста */
      border: 1px solid #666; /* Параметры рамки */
      background: #eee; /* Цвет фона */
      padding: 5px; /* Поля вокруг текста */
    }
  </style>
</head>
<body>
  <p>Обычный параграф</p>
  <p id="ора">Параграф необычный</p>
</body>
</html>

```

Результат данного примера показан на рис. 9.2.

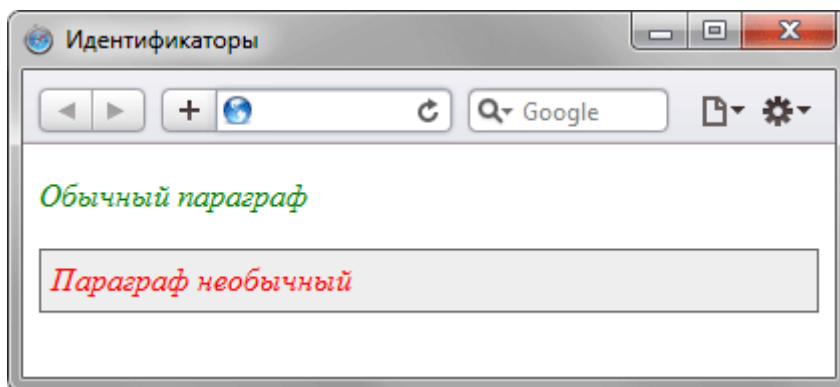


Рис. 9.2. Вид текста после применения стиля

В данном примере вводится стиль для тега `<p>` и для такого же тега, но с указанием идентификатора `ора`.



## Часть 2

### Ход работы:

1. Создаём документ с именем 2020.html и вставляем туда основную часть документа:

```
<html>
```

```
</html>
```

```
<head>
```

```
</head>
```

```
<body>
```

```
</body>
```

```
</html>
```

2. В головной части задаём заголовок страницы и конструкцию использования каскадных таблиц, которые мы расположим в документе main.css.

`<div></div>` используется для логического выделения блока HTML-документа.

Элемент группировки, как и элемент SPAN. В современном сайтостроении используется как удобный контейнер для объектов страницы, которым легко динамически манипулировать - перемещать, включать/выключать, создавать слои, регулировать отступы и т.п.

В браузеронезависимой вёрстке обычно используется для выравнивания блока html-кода в окне браузера.

Находящиеся между начальным и конечным тегами текст или HTML-элементы по умолчанию оформляются как отдельный параграф.

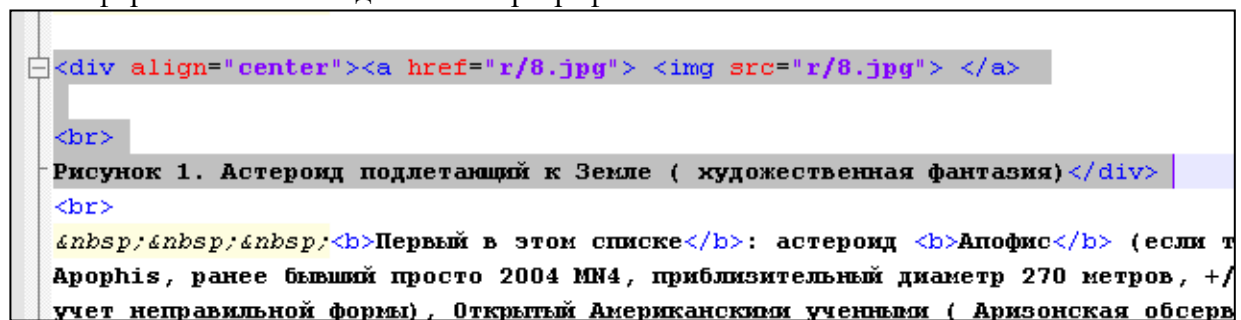


Рис.1 «Элемент div»

На рисунке видно, что ссылка на картинку и подпись к ней расположены в `<div>` и расположены по центру.

3. Текст заключаем в параграф `<p></p>`. Для пробелов используем специальный символ «`&nbsp;`». Для перехода на другую строку необходимо ставить в том месте



```

table{
border-style:double;
border-color:white;
border-collapse:separate;
text-align: center;
background-color: silver;
border-spacing: 3px 3px;
font-size: 110%;
}

```

Рис.5 «Параметры таблицы»

Вид ячеек также задаём в том же документе:

```

td{
border-style:double;
border-color:white;
border-collapse:separate;
background-color: silver;
}

```

Рис.6 «Параметры ячеек»

## 7. Вид готовой страницы с таблицей

Как видно на схеме более 70% вероятности падения в океан и трудно доступные районы Сибири - 20%.

Таблица прогнозируемых последствий падения.

| Последствия падения на расстоянии от эпицентра(км) | 10                                                                        | 50                                                     | 120                                                    |
|----------------------------------------------------|---------------------------------------------------------------------------|--------------------------------------------------------|--------------------------------------------------------|
| Сила землетрясения (по шкале Рихтера)              | 6,5                                                                       | 5,6                                                    | 4,9                                                    |
| Скорость ветра(ударная волна) м/с                  | 792                                                                       | 77,8                                                   | 44,7                                                   |
| Разрушения                                         | Обрушение укрепленных зданий, обрушение туннелей в метро, трещины в земле | Разрушение неукрепленных зданий, разрывы трубопроводов | Падение мебели, штукатурки, незначительные последствия |
| Состояние населения(жертвы,%)                      | 99,0                                                                      | 70,0                                                   | 5,0                                                    |

Анализируя таблицу, видим, только попад в центр большого города, астероид может вызвать серьезные людские жертвы, а это весьма маловероятно. Взрыв термоядерной бомбы намного страшнее. И даже при этом огромнейшие разрушения.

Рис.7 «Таблица»

Задание для самостоятельной работы:

- IV. Создайте 2 собственных HTML документа – описание двух ваших товаров или услуг, содержащие фоновый рисунок страницы, элементы

форматирования текста, нумерованный и ненумерованный списки, ссылки для скачивания файлов. Для работы использовать только редактор Notepad++ и браузер.

- V. Создайте собственный HTML документ – прайс - лист ваших товаров или услуг, содержащий фоновый рисунок страницы, элементы форматирования текста, таблицу, рисунки в ячейках таблицы, ссылки для перехода на страницы, созданные в первой части лабораторной работы, на этих страницах добавить ссылки для возврата на страницу с прайс листом.
- VI. Разместите созданные страницы в отдельной папке своего раздела хостинга, на стартовой странице в ячейке «Ссылка 5» разместите ссылку на созданную таблицу со списком товаров.
- VII. Для работы использовать только редактор Notepad++ и браузер.
- VIII. Оформить отчёт, в котором представить описание хода выполнения работы и ответы на контрольные вопросы.

### **Контрольные вопросы**

1. В чем отличие CSS3?
2. Что такое селекторы?
3. Какими свойствами обладают селекторы потомков?
4. Что такое селекторы дочерних элементов?
5. Как проявляют себя селекторы элементов одного уровня?
6. Что такое псевдоклассы?
7. Как проявляют себя псевдоклассы дочерних элементов?
8. Что такое псевдо-классы форм?
9. Что такое псевдоэлементы?
10. Для чего применяют селекторы атрибутов?
11. В чем сущность наследования стилей?
12. В чем сущность каскадности стилей?