

Применение каскадных таблиц стилей (CSS)

Лекции	Описание
1. Назначение и применение CSS	Основные понятия CSS, их назначение, определение и использование при форматировании HTML-документа.
2. Блочные и строковые элементы	Блочные и строковые элементы: описание, форматирование и свойства.
3. Цвет и шрифт	Управление отображением цветами текста и фоном, на котором отображается текст, а также использование гарнитур шрифтов.
4. Текст и списки	Свойства текстовых фрагментов: межбуквенные расстояния, высота строк, выравнивание, отступ в первой строке параграфа, преобразования начертания. Управление формой и отображением списков.
5. Позиционирование	Размещение блочных элементов HTML-разметки в рабочей области браузера с точностью до пикселя: размеры блока, абсолютные и относительные координаты. Слои: управление видимостью.
Литература	

1: Назначение и применение CSS

Назначение CSS

Дизайн Web-узлов — это точное размещение компонентов HTML-страниц относительно друг друга в рабочей области окна браузера.

Недостатки такого определения Web-дизайна очевидны. В нем не учтены ни цвет, ни форма, ни другие свойства компонентов HTML-страниц. Главное в этом определении — показать ограниченность возможностей HTML-разметки. Позиционирование компонентов на странице является одним из самых слабых мест в HTML.

К компонентам страницы относятся: блоки текста, графика и встроенные приложения. Размер и границы каждого из этих компонентов в рамках HTML-разметки задаются с разной степенью точности. Размер графики и приложений можно задать с точностью до пиксела. Размеры текстовых блоков в HTML задать нельзя: они вычисляются браузером исходя из относительного размера шрифта по умолчанию.

Автор страницы не может заранее определить настройки браузера пользователя, что существенно ограничивает число вариантов представления информации на странице.

Нельзя сказать, что разработчики браузеров не пытались изменить данную ситуацию. В ранних версиях браузеров CERN для платформы NEXT и в браузерах WWWC автор страницы имел возможность переопределять настройки браузера по умолчанию через HTML-разметку. Но этот подход не получил продолжения в коммерческих продуктах.

Другой способ управления настройками браузера — программирование на JavaScript. Бурное развитие этого языка позволяет говорить о возможности полного контроля над процессом отображения HTML-страниц. Недостаток JavaScript — отказ от декларативного характера разметки и относительно большой объем кода для переопределения свойств элементов разметки.

Спецификация CSS (Cascading Style Sheets) позволяет остаться в рамках декларативного характера разметки страницы и полностью контролировать

форму представления элементов HTML-разметки.

Каскадные таблицы стилей призваны разрешить противоречие между точностью определения размеров картинок и приложений, с одной стороны, и точностью определения размеров блоков текста и его начертания — с другой.

Таблицы стилей также позволяют определить цвет и начертание текстового фрагмента, изменять эти параметры внутри текстового блока, выполнять выравнивание текстового блока относительно других блоков и компонентов страницы.

Наличие подобных возможностей позволяет говорить о CSS как о средстве разделения логической структуры документа и формы его представления. Логическая структура документа определяется элементами HTML-разметки, в то время как форма представления каждого из этих элементов задается CSS-описателем элемента.

CSS позволяет полностью переопределить форму представления элемента разметки по умолчанию. Например, `<I>...</I>` определяет отображение текста курсивом:

```
<I>Отообразим текст курсивом</I>
```

А теперь переопределим стиль отображения для элемента разметки `I` ([открыть](#)):

```
<I >  
Отообразим текст курсивом  
</I>
```

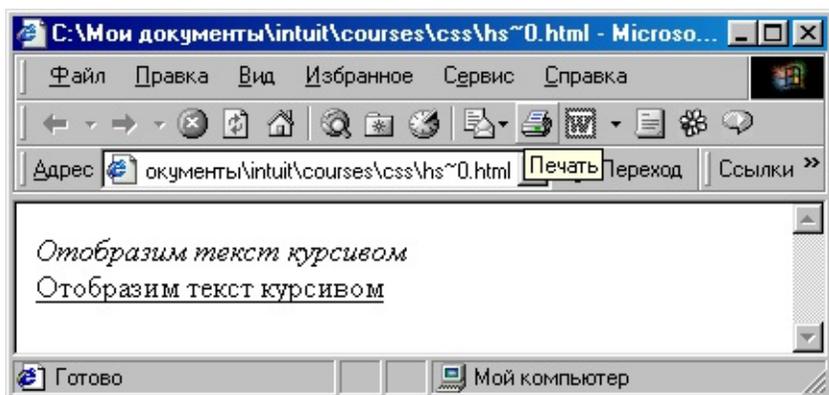


Рис. 8.1.

Этот пример показывает, что привычный стиль отображения элементов может быть полностью изменен при помощи CSS. В данной технологии HTML-разметка носит чисто декларативный характер.

Практическое значение CSS для Web-инжиниринга (совокупности технологий разработки и сопровождения Web-узлов) заключается в том, что процесс создания узла можно формализовать и представить в виде последовательности действий:

- необходимо определиться с номенклатурой страниц, т.е. все страницы проектируемого Web-узла разбить на типы, например домашняя страница, навигационные страницы, информационные страницы, коммуникационные страницы и т.п. У каждого узла этот перечень может быть свой;
- для каждого типа страниц требуется разработать определенную логическую структуру (стандартный набор компонентов страницы);
- следует создать навигационную карту узла и форму ее реализации на страницах;
- для каждого стандартного компонента страницы нужно разработать стиль его отображения (CSS-описатель);
- теперь остается только рисовать картинки, создавать анимацию, писать программы, вручную вводить текст и графику или генерировать содержимое страниц автоматически при обращении к ним.

Объяснив таким образом роль и назначение CSS среди многообразия Web-технологий, мы переходим непосредственно к обсуждению применения каскадных таблиц стилей.

Способы применения CSS

Под способами применения CSS мы в данном разделе понимаем форму декларирования стиля на HTML-странице и форму связывания описания стиля отображения элемента разметки с самим элементом. Речь идет о том, где и в какой форме автор страницы (или дизайнер) описывает стиль, и как и в какой форме на него ссылается.

Итак, различают четыре способа применения стилей:

- переопределение стиля в элементе разметки;

- размещение описания стиля в заголовке документа в элементе STYLE;
- размещение ссылки на внешнее описание через элемент LINK;
- импорт описания стиля в документ.

Здесь мы следуем за Джорджем Янгом из Microsoft (Cascading Style Sheets in Internet Explorer 4.0. Microsoft, 1997). Важно отметить, что в браузерах других производителей импорт стиля не поддерживается. Однако, поскольку патент на CSS1 принадлежит Microsoft, опустить импорт в нашем описании было бы неправильно.

Переопределение стиля

```
<h1 >  
Заголовок первого уровня  
</h1>
```

Атрибут style можно применить внутри любого элемента разметки. Например, мы можем через style определить ширину и выравнивание элемента hr (горизонтальное отчеркивание):

```
<hr >
```

Очевидно, что не все параметры стиля можно установить для конкретного элемента разметки. О типах элементов и соответствующих параметрах стилей мы поговорим в разделе "Понятия блочного и строкового элементов".

Здесь же нужно отметить следующее: стили разработаны в первую очередь для управления отображением текста. Не следует увлекаться стилями при управлении отображением нетекстовых элементов HTML-разметки.

Элемент STYLE

Применение элемента STYLE — это основной способ внедрения каскадных таблиц стилей в ткань HTML-документа. Помимо управления отображением элементов разметки, элемент STYLE позволяет описывать стилевые свойства элементов, которые можно изменять при программировании на JavaScript.

Элемент STYLE дает возможность определить стиль отображения для:

- стандартных элементов HTML-разметки;
- произвольных классов (селектор CLASS);
- HTML-объектов (селектор ID).

К сожалению, работа с селекторами в браузерах различных производителей может преподносить различного рода сюрпризы. Особенно это касается селектора ID. В данном случае мы будем рассматривать Microsoft как держателя патента на спецификацию CSS.

Понятие селектора, применение селекторов и формальный синтаксис CSS мы обсудим в разделах "Синтаксис" и "Наследование и переопределение".

Стандартные элементы разметки описываются в элементе STYLE следующим образом:

```
<HEAD>
<STYLE>
p { color:darkred;text-align:justify;
    font-size:8pt; }</STYLE>
</HEAD>
<BODY>
...
<P>
Этот параграф мы используем как пример
применения описания стиля для стандартного
элемента HTML-разметки.
</P>
...
</BODY>
```

Теперь все параграфы документа будут отображаться стилем из элемента STYLE, если только стиль не будет каким-либо способом переопределен. В STYLE можно определить стиль любого элемента разметки.

Ссылка на внешнее описание

Ссылка на описание стиля, расположенное за пределами документа, осуществляется при помощи элемента LINK, который размещают в элементе HEAD. Внешнее описание может представлять собой файл, содержащий описание стилей. Описание стилей в этом файле будет по синтаксису в точности совпадать с содержанием элемента STYLE.

Ниже приведен пример ссылки на внешнее описание стилей:

```
<LINK TYPE="text/css" REL="stylesheet"
      HREF="http://intuit.ru/my_css.css">
```

Здесь важны значения атрибутов REL и TYPE. Атрибут REL должен иметь значение stylesheet. Type может принимать значения text/css или text/javascript. Второй тип описания стилей введен Netscape. Его мы в данном учебном курсе не обсуждаем.

Атрибут HREF задает универсальный локатор ресурса (URL) для внешнего файла описания стилей. Это может быть ссылка на файл с любым именем, а не только на файл с расширением *.css.

Импорт описания стилей

Импорт описателей стилей в некотором смысле составляет конкуренцию представленному выше указанию на внешний описатель стиля.

Импортировать стиль можно либо внутрь элемента STYLE, либо внутрь внешнего файла, который представляет собой описатель стиля. Оператор импорта стиля должен предшествовать всем прочим описателям стилей:

```
<STYLE>
@import:url(http://intuit.ru/style.css)
A { color:cyan;text-decoration:underline; }
</STYLE>
```

Импортируемый стиль можно переопределить либо через описатель элемента в STYLE, либо через атрибут элемента STYLE.

Синтаксис

Формально стиль отображения элементов разметки задается ссылкой в элементе разметки на селектор стиля. Синтаксис описания стилей в общем виде представляется следующим образом:

```
selector[, selector[, ...]]
  { attribute:value;
    [attribute:value;...] }
```

или

```
selector selector [selector ...]
  { attribute:value;
    [attribute:value;...] }
```

В первом варианте перечислены селекторы, для которых действует данное описание стиля. Второй вариант задает иерархию вложенности селекторов, для совокупности которых определен стиль. Напомним, что речь в данном случае идет об описаниях стилей в нотации text/css. Описания стилей размещаются либо внутри элемента STYLE, либо во внешнем файле.

В качестве селектора можно использовать имя элемента разметки, имя класса и идентификатор объекта на HTML-странице.

Атрибут (attribute) определяет свойство отображаемого элемента, например левый отступ параграфа (margin-left), а значение (value) — значение этого атрибута, например, 10 типографских пунктов (10 pt).

Селектор — имя элемента разметки

Когда автор Web-узла хочет определить общий стиль всех страниц, он просто прописывает стили для всех элементов HTML-разметки, которые будут использоваться на страницах. Это дает возможность компоновать страницы из логических элементов, а стиль отображения элементов описать во внешнем файле.

Такой способ создания сайта позволяет автору изменять внешний вид всех страниц путем внесения изменений в файл описания стилей, а не в файлы HTML-страниц.

Внешний файл при этом может выглядеть следующим образом:

```
I, EM {color:#003366;font-style:normal}
A I {font-style:normal;font-weight:bold;
    text-decoration:line-through}
```

В первой строке этого описания перечислены селекторы-элементы, которые будут отображаться одинаково:

<I>это курсив</I> и это тоже курсив

Последняя строка определяет стиль отображения вложенного в гипертекстовую ссылку курсива:

```
<A NAME=empty><I>intuit</I></A>
```

В данном случае переопределение состоит в том, что текст отображается внутри гипертекстовой ссылки перечеркнутым, причем жирным шрифтом.

Селектор — имя класса

Имя класса не является каким-либо стандартным именем элемента HTML-разметки. Оно определяет описание класса элементов разметки, которые будут отображаться одинаково. Для того, чтобы отнести элемент разметки к тому или иному классу, нужно воспользоваться его атрибутом CLASS ([открыть](#)):

```
<STYLE>
.test {color:white;background-color:black;}
</STYLE>
...
<P CLASS="test">
Этот параграф мы отобразим белым цветом по
черному фону
</P>
...
<P>
Эту <A CLASS="test">гипертекстовую ссылку</A>
мы отобразим белым цветом по черному фону.
</P>
```

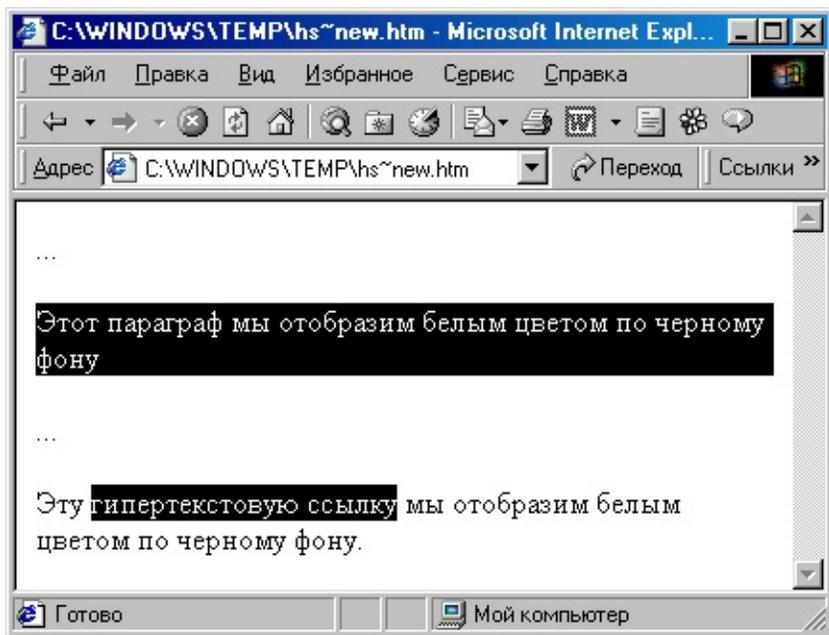


Рис. 8.2.

Таким образом в любом элементе разметки можно сослаться на описание класса отображения. При этом совершенно необязательно, чтобы элементы разметки были однотипными. В примере к одному классу отнесены и параграф, и гипертекстовая ссылка в другом параграфе.

Лидирующую точку в имени класса можно опустить. Она задается из соображений сохранения единства описания. Например, можно определить классы отображения однотипных элементов разметки:

```
a.menu { color:red;background-color:white;
         text-decoration:none; }
a.paragraph { color:navy;
              text-decoration:underline; }
```

В данном примере класс гипертекстовых ссылок `menu` имеет одно описание стиля, а класс гипертекстовых ссылок `paragraph` — совершенно другое. При этом каждый из этих классов нельзя применить к другим элементам разметки, например, параграфу или списку. Если имя элемента разметки не задано, это означает, что класс можно отнести к любому элементу разметки — корневой класс описания стилей. Это очень похоже на обозначение имени корневого домена в системе доменных имен. Собственно ничего удивительного здесь нет, т.к. система классов объектов на HTML-странице представляет собой дерево. Элементы разметки — это узлы дерева.

Селектор — идентификатор объекта

Объектная модель документа (Document Object Model) описывает документ как дерево объектов. Объектами являются: сам документ, его разделы (элемент DIV), картинки, параграфы, приложения и т.п. Каждый из объектов можно поименовать и обращаться к нему по имени. Данная возможность используется при программировании страниц на стороне клиента.

Применение идентификатора объекта оправдано еще и в случае модификации атрибута описания стиля для данного объекта в его CSS-описании. Вместо двух описаний классов, которые отличаются только одним из параметров, можно создать одно описание класса и описание идентификатора объекта. Описание стиля для объекта задается строкой, в которой селектор представляет собой имя этого объекта с лидирующим символом "#":

```
a.mainlink { color:darkred;
             text-decoration:underline;
             font-style:italic; }
#blue { color:#003366 }
...
<A CLASS=mainlink>основная гипертекстовая
ссылка</A>
<A CLASS=mainlink ID=blue>модифицированная
гипертекстовая ссылка</A>
```

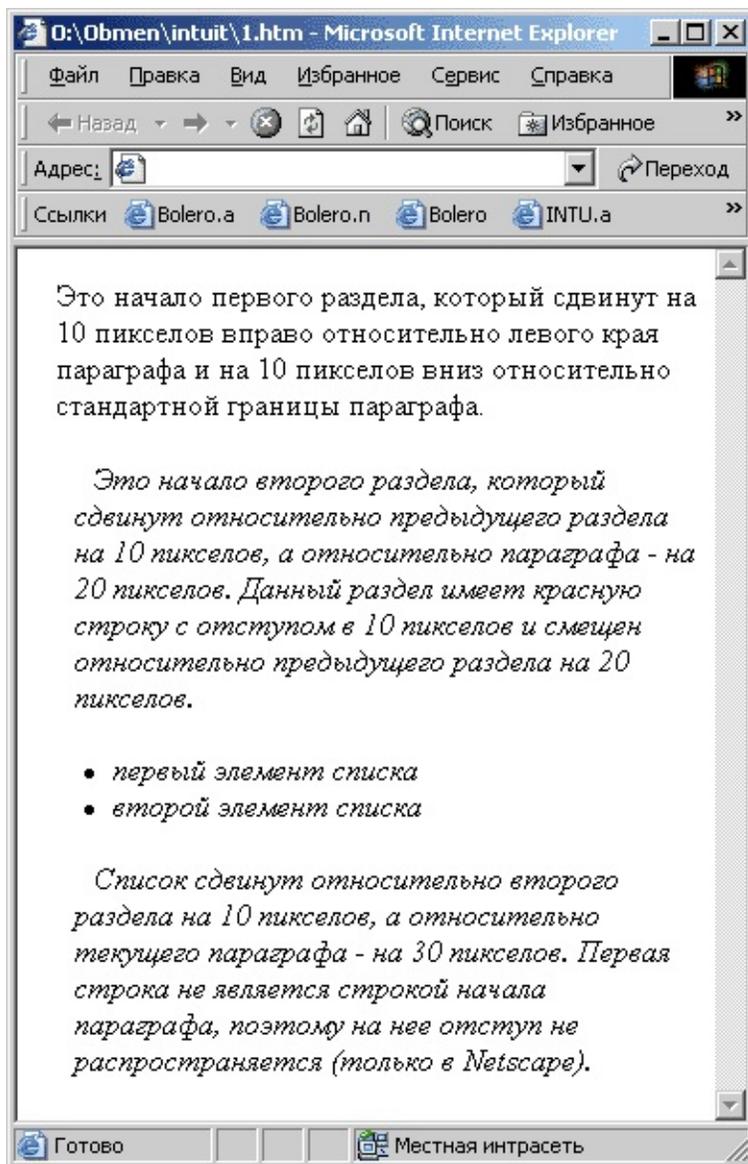
Следует отметить, что интерпретация идентификаторов объектов в Internet Explorer и Netscape Navigator различна. Существует еще атрибут name у элемента разметки. При идентификации объекта Netscape Navigator обычно имеет дело именно с этим атрибутом, а Internet Explorer — с атрибутом ID.

Различия в интерпретации ID в браузерах при декларативном использовании CSS не очень страшны. Другое дело, если автор решится программировать стили, т.е. изменять значения атрибутов описателей стилей. В этом случае разница объектных моделей документов в Netscape Navigator и Internet Explorer проявится в полной мере. Фактически, придется для каждого из браузеров разрабатывать совершенно разные страницы.

Наследование и переопределение

При обсуждении технических спецификаций часто бывает полезно вникнуть в смысл названия. В названии принято точно определять суть и назначение стандарта или спецификации. Описание стилей отображения элементов HTML-разметки носит название "Каскадные таблицы стилей". Со словом "стилей" все более-менее понятно. Под словом "таблицы" следует понимать набор свойств элемента разметки, который можно представить в виде строки в таблице свойств, т.е. элементы разметки — строки, а свойства — столбцы. А вот слово "каскадные" требует пояснения.

Во-первых, существует иерархия элементов разметки (дерево объектов на странице). Во-вторых, свойства этих объектов могут наследоваться. Таким образом в дереве объектов образуется ветвь, которая ведет к листу дерева — элементу разметки, например, элементу списка или параграфу. Его свойства определяются элементами разметки, в которые вложен элемент, и описателями стиля для данного элемента:



Предыдущий текст закодирован в терминах разделов и списка следующим образом:

<DIV >

Это начало первого раздела, который сдвинут на 10 пикселей вправо относительно левого края параграфа и на 10 пикселей вниз относительно стандартной границы параграфа.

<DIV >

Это начало второго раздела, который сдвинут относительно предыдущего раздела на 10 пикселей, а относительно параграфа – на 20 пикселей. Данный раздел имеет красную строку

строку с отступом в 10 пикселей и смещен относительно предыдущего раздела на 20 пикселей.

```
<UL >  
<LI>первый элемент списка  
<LI>второй элемент списка  
</UL>
```

Список сдвинут относительно второго раздела на 10 пикселей, а относительно текущего параграфа – на 30 пикселей. Первая строка не является строкой начала параграфа, поэтому на нее отступ не распространяется (только в Netscape).

```
</DIV>  
</DIV>
```

Таким образом отступы отсчитываются относительно элемента, в который вложен текущий элемент. Все параметры, которые не были переопределены в текущем элементе, наследуются из старшего по иерархии элемента. Последнее хорошо продемонстрировано в применении стилей отображения списка, который вложен в раздел и поэтому отображается курсивом.

Когда объяснение некоторого феномена HTML-разметки растягивается на несколько параграфов, имеет смысл воспользоваться приведенной ниже графической схемой построения страницы.

При использовании стилей действуют следующие правила старшинства стилей:



Рис. 8.3.

- сначала применяются стили браузера по умолчанию;
- стили браузера по умолчанию переопределяются прилинкованными стилями (элемент LINK заголовка документа);
- прилинкованные стили переопределяются описаниями стилей в элементе STYLE;

- стили элемента STYLE переопределяются атрибутом STYLE в любом из элементов разметки.

Не все атрибуты стиля могут наследоваться. Например, "набивка" (отступ содержания элемента от его границ) элемента BODY не наследуется вложенными в него элементами и определяется по умолчанию или прописывается для каждого элемента отдельно. Алгоритмы наследования в Internet Explorer и в Netscape Navigator разные, поэтому для единства отображения элементов следует прописывать стиль по максимуму атрибутов.

2: Блочные и строковые элементы

Блочные и строковые элементы

В описании элементов разметки языка HTML существует понятие строкового (in-line) элемента разметки и блочного (block) элемента разметки. Формально они определены в DTD SGML-описания языка HTML. Объяснить различие между блочным и строковым элементами можно на примере:

- **параграф** — это блочный элемент разметки;
- **выделение курсивом** — это строковый элемент разметки.

Блочные элементы можно вкладывать друг в друга, но они не должны пересекаться. Строковые элементы можно как вкладывать, так и пересекать (согласно DTD и практике старых версий браузеров), но последнее делать не рекомендуется.

Очевидно, что по набору атрибутов управления отображением (атрибуты описания стиля) строковые и блочные элементы отличаются. Упрощенно можно сказать, что атрибуты описания стиля строкового элемента являются подмножеством атрибутов описания стиля блочного элемента.

Обобщениями блочного и строкового элементов, с точки зрения стилей, являются элементы DIV и SPAN, соответственно.

Элемент DIV

DIV играет роль универсального блока. Блочный элемент всегда отделен от прочих элементов страницы (контекста) пустой строкой. DIV не несет никакой смысловой нагрузки. Часто говорят, что DIV — это раздел страницы. Но на самом деле его применение имеет смысл только в контексте CSS. Никаких правил по умолчанию для отображения DIV не существует. Это просто новая строка текста.

DIV позволяет применить атрибуты стиля, связанные с границей блока и отступами блока от границ старшего элемента, а также "набивку", т.е.

отступ от границы блока до границы вложенного элемента: ([открыть](#))

<DIV >

Блочный элемент, заданный элементом разметки DIV.

<P>Для него определена граница и отступы как от границ старшего элемента разметки, так и для вложенных в него элементов разметки.</P>

</DIV>

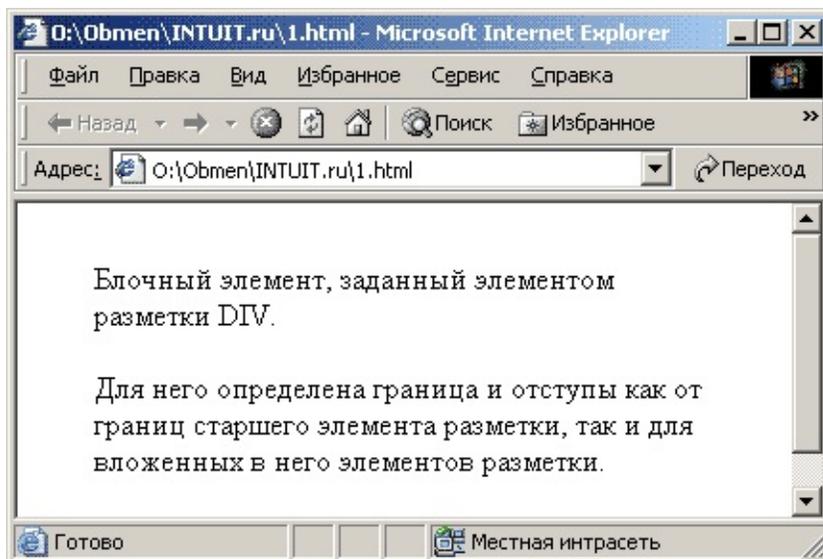


Рис. 9.1.

В данном примере внутри окна браузера расположен блочный элемент (DIV), внутрь которого помещен еще один блочный элемент(P). DIV имеет белый фон и границу.

Если текст будет просматриваться браузерами, не поддерживающими CSS, элемент DIV использовать не рекомендуется. В этом случае лучше применить параграф или другой подходящий по смыслу элемент разметки из стандартного набора HTML.

Элемент SPAN

Элемент разметки SPAN — это обобщенный строковый элемент разметки, применение которого не приводит к образованию блока текста. Он может заменить элементы FONT, I, B, U, SUB, SUP и т.п. Приведем примеры таких соответствий:

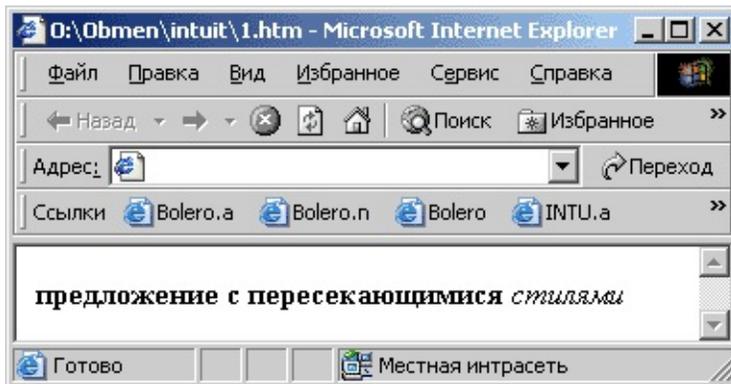
Таблица 9.1.

HTML-элемент	CSS-аналог

<I>...</I>	...
...	...
<U>...</U>	...
	и т.п.

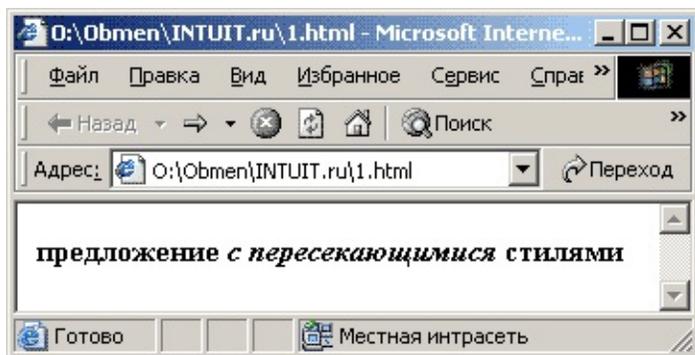
В новых версиях браузера Netscape описания строковых стилей пересекаться не должны. Тэг конца элемента строкового типа закрывает ближайший элемент, а не тот, который открыт тэгом начала данного строкового стиля. Также и в случае применения элемента SPAN, где тэг конца можно соотнести только с ближайшим тэгом начала элемента SPAN:

предложение <I>с пересекающимися
стилями</I>



предложение

с пересекающимися стилями



Применение элемента SPAN ограничено браузерами, которые поддерживают CSS. При этом не все атрибуты спецификации CSS поддерживаются в браузерах. Например, атрибут vertical-align, который призван заменить элементы SUP и SUB, не поддерживается ни одним из браузеров.

3: Цвет и шрифт

Управление цветом в CSS

Каскадные таблицы стилей (CSS) в первую очередь описывают свойства текста. Это касается как текстовых блоков, так и строковых элементов разметки содержания страницы. В данном разделе речь пойдет об управлении отображением цвета текста (color) и цвета фона (background-color), на котором отображается текст.

Кроме цвета текста и цвета фона CSS позволяет определять цвет границы текстового блока (border-color).

Атрибуты стилей, которые мы собираемся рассмотреть, согласно спецификации Microsoft, относятся к группе атрибутов Color and Background Properties. Всего в эту группу входит семь атрибутов, шесть из которых определяют свойства фона. Кроме цвета фона и его прозрачности, можно управлять фоновой картинкой (координатами ее размещения и способами повторения). К сожалению, Netscape Navigator большинство из этих атрибутов не поддерживает, поэтому мы не будем рассматривать их подробно.

Интерпретация атрибутов цвета в Netscape Navigator и Internet Explorer различна. В Netscape Navigator фоновый цвет отображается только там, где есть текст, а в Internet Explorer фоновый цвет заливает весь блок или строковый элемент вне зависимости от наличия в нем текста.

Цвет текста

В HTML для управления цветом отображаемого текста используется элемент FONT. Его аналогом в CSS является атрибут color. Этот атрибут можно применять как для блочных, так и для строковых элементов разметки.

Рассмотрим в качестве блочного элемента разметки ячейку таблицы:
([открыть](#))

```
TD { color:darkred; }
```

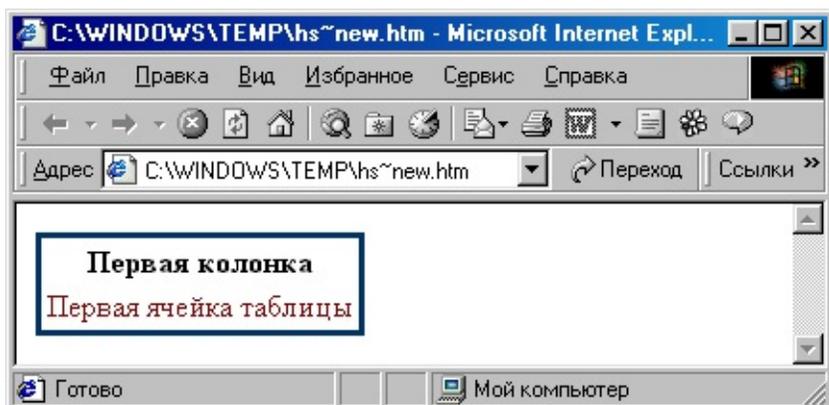


Рис. 10.1. Блочный элемент разметки для ячейки таблицы.

В данном примере цвет текста определен только для обычной ячейки, поэтому содержание заголовка колонки отображается основным цветом (#003366).

При определении цвета текста для блочного элемента весь текст этого элемента отображается заданным цветом. Частичное изменение цвета возможно, если поместить строковый элемент разметки внутри блочного: ([открыть](#))

```
P { color:darkred; }  
I { color:#003366;font-style:normal; }
```

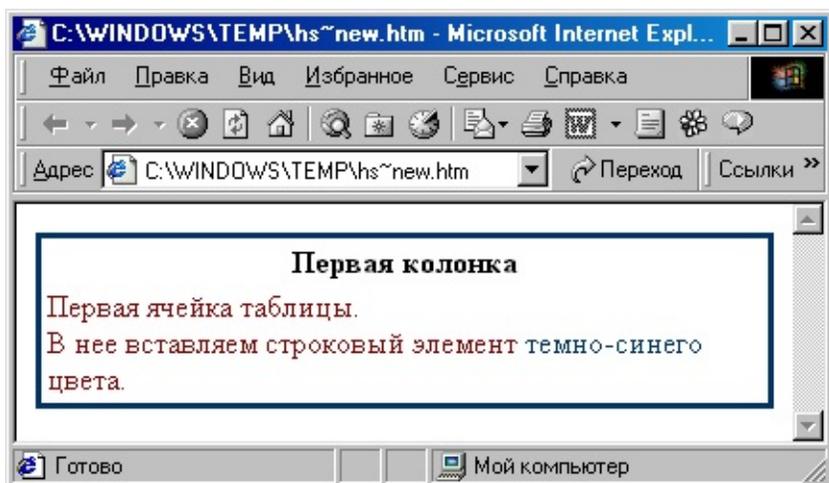


Рис. 10.2. Частичное изменение цвета текста внутри блочного элемента

В данном примере в качестве блочного элемента мы используем параграф, а в качестве строкового элемента (in-line) применяем I. Таблица в данном случае большого значения не имеет, но используется для единообразия с предыдущим примером. В нее мы помещаем параграф со встроенным в него in-line элементом разметки.

Цвет фона текста

В HTML цветом фона можно управлять только для конкретного блочного элемента разметки. Таким элементом может быть вся страница:

```
<BODY BGCOLOR=...>...</BODY>
```

Или, например, таблица:

```
<TABLE BGCOLOR=...>...</TABLE>
```

В приведенном ниже примере для выделения текста применено инвертирование цвета фона и цвета текста:([открыть](#))

```
<SPAN  
>  
как строковые элементы разметки  
</SPAN>
```

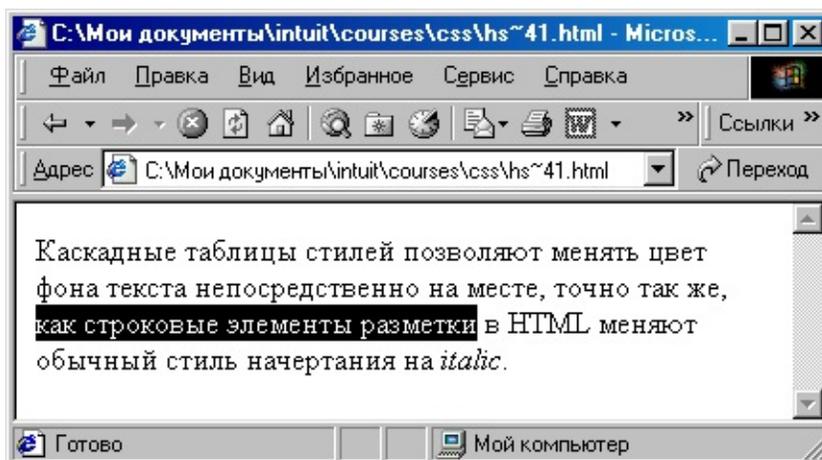


Рис. 10.3. Инвертирование цвета фона и цвета текста для выделения текста.

При использовании цвета фона следует помнить, что поддержка этого атрибута реализована для всех блочных элементов разметки только в Internet Explorer 4.0. Поддержка CSS в версиях Netscape Navigator гораздо скромнее.

Для работы с фоном элементов существует несколько атрибутов, которые поддерживаются только в Internet Explorer, начиная с версии 4.0: background-image; background-repeat; background-attachment; background-position. Все свойства фона можно описать в атрибуте background:

```
background:transparent|color url repeat  
          scroll position
```

Пример:

```
P { background: gray  
    http://intuit.ru/intuit.gif  
    no-repeat fixed center center; }
```

Однако при всем изобилии возможностей, злоупотреблять ими не стоит.

Шрифт

Шрифтам в компьютерной графике всегда уделялось много внимания, и World Wide Web не является исключением. Но все богатство и разнообразие существующих шрифтов для русского языка ограничено фактически тремя шрифтами: serif (обычно Times или другой шрифт с засечками), sans-serif (Arial, Helvetica или другой шрифт без засечек) и monospace (Courier). Если быть точным, то здесь перечислены семейства шрифтов. Обычно каждое из этих семейств представлено только одним кириллическим шрифтом.

Автор документа для управления отображением букв может применить несколько атрибутов, влияющих на шрифт:

font-family - семейство начертаний шрифта (гарнитура);

font-style - прямое начертание или курсив;

font-weight - "усиление" (насыщенность) шрифта, "жирность" букв;

font-size - размер шрифта (кегель). Задается в пикселах (px) и типографских пунктах (pt).

font-variant - вариант начертания (обычный или мелкими буквами - капитель).

Все эти параметры можно совместить в одном атрибуте font:

```
font:bold 12pt sans;
```

Правда, нет никакой уверенности в том, что последнее определение шрифта будет работать во всех браузерах.

При использовании различных гарнитур (font-family) следует помнить, что наличие или отсутствие необходимой автору гарнитуры всецело зависит от предпочтений пользователя. Для кириллицы это может вылиться в появление абракадабры там, где автор применяет отсутствующие у пользователя шрифты.

Самое неприятное, с чем можно столкнуться при использовании шрифтов - это несоответствие моноширинных шрифтов, которые применяются в HTML-формах. Обратная связь с пользователем в этом случае невозможна.

Спецификация CSS предусматривает перечисление шрифтов в описаниях стилей, что позволяет частично решить проблему подбора шрифта. К сожалению, в Unix и Windows шрифты не согласованы. Фактически, при разработке страниц в CSS используются только классы шрифтов (serif, sans-serif и monospace).

Гарнитура (font-family)

Гарнитура шрифта - это набор начертаний одного шрифта. Шрифт может иметь "прямое" начертание (normal), курсив (italic), "скошенное" (oblique), усиленное по насыщенности ("жирное", bold), "мелкое" (капитель, small-caps) и т.п.

Наиболее распространенные гарнитуры в российской части Web - это

Times, Arial, Courier. Причем все они принадлежат к разным группам шрифтов. Times - это пропорциональный шрифт "с засечками" (serif), Arial - это пропорциональный шрифт "без засечек" (sans-serif), а Courier - это моноширинный шрифт (monospace). В Unix вместо Arial чаще применяется Helvetica.

В чем разница между этими группами шрифтов, можно показать на примере:([открыть](#))

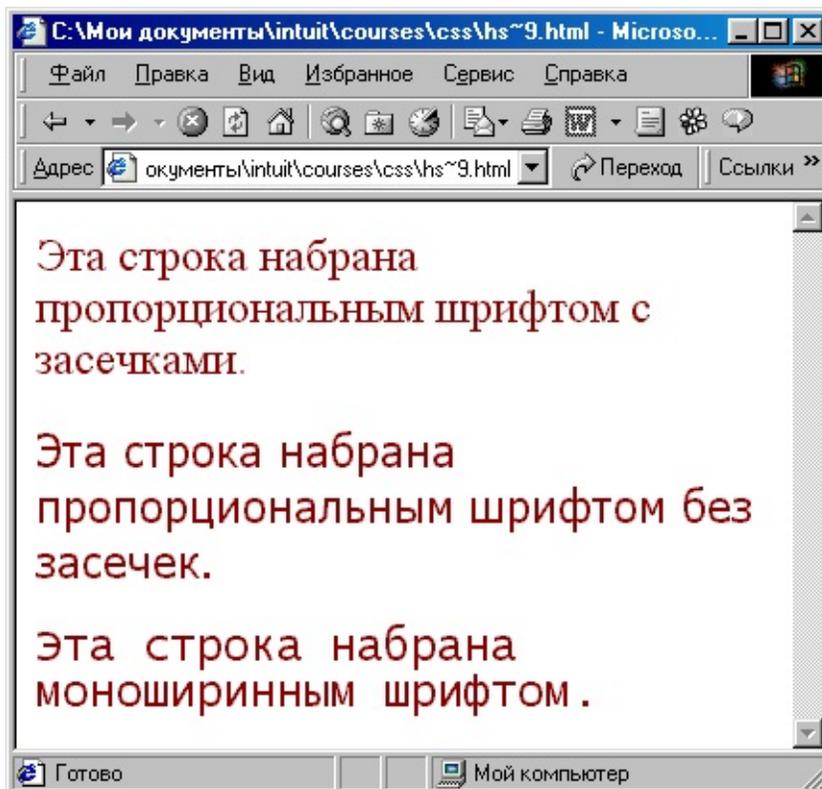


Рис. 10.4. Демонстрация групп шрифтов.

```
<P ALIGN=left >  
Эта строка набрана пропорциональным шрифтом  
с засечками.</P>  
<P ALIGN=left >  
Эта строка набрана пропорциональным шрифтом  
без засечек.</P>  
<P ALIGN=left >  
Эта строка набрана моноширинным шрифтом.</P>
```

При указании имени группы шрифтов, как показано выше, браузер

подбирает подходящий для отображения шрифт данной группы из имеющегося набора шрифтов.

Если оптимизация браузера автора страницы не устраивает, можно указать непосредственно имя гарнитуры шрифта:([открыть](#))

```
<SPAN >  
N</SPAN><BR>  
<SPAN >  
A<SUB>x, y</SUB>=e (y<SUB>a</SUB>+x<SUB>a</SUB>)  
</SPAN><BR>  
<SPAN >a=1  
</SPAN>
```

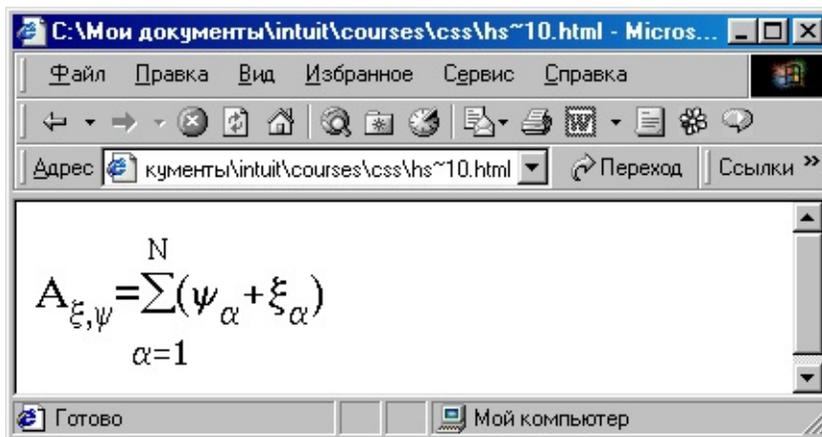


Рис. 10.5. Прямое указание гарнитуры шрифта.

Если в вашей операционной системе поддерживается шрифт symbol, то вы увидите нормально записанную формулу суммы, если же нет - абракадабру. Однако в данном случае применение обычного шрифта вообще бессмысленно - в нем формула не будет отображена правильно.

Теперь рассмотрим менее экзотический пример. Речь пойдет о тексте в поле ввода формы.

В первом случае мы применяем просто шрифт по умолчанию. Далее изменим его на другой моноширинный шрифт:([открыть](#))

```
<b>шрифт по умолчанию:</b><br>  
<INPUT VALUE="Русский язык">  
<b>моноширинный шрифт:</b><br>
```

```
<INPUT VALUE="Русский язык">  
<INPUT VALUE="Русский язык">
```

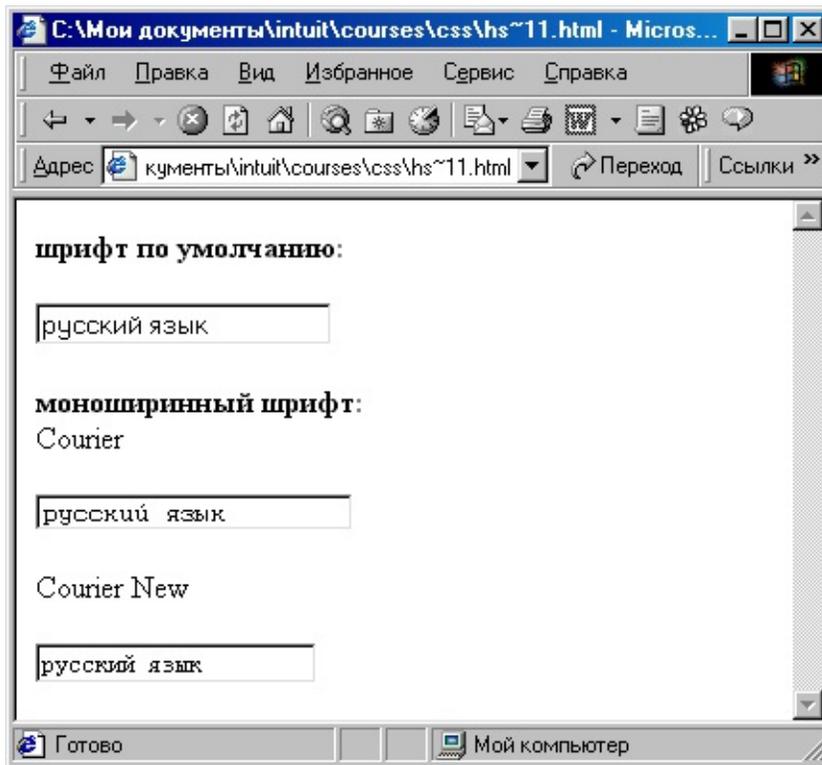


Рис. 10.6. Смена шрифтов в форме ввода.

В операционной системе, в которой тестировались примеры для этой работы, нет кириллицы в гарнитуре Courier, но зато она есть в гарнитуре Courier New. Следовательно, тот пример, в котором применяется гарнитура Courier, будет у нас отображаться абракадаброй.

Если пользователь работает с другим набором шрифтов, результат будет иным.

Данный пример со всей очевидностью показывает, что при выборе шрифта, а точнее, гарнитуры шрифта, не нужно мудрить. Лучше всего пользоваться категориями групп шрифтов, тогда текст с точки зрения языка будет везде отображаться правильно. Правда, достигаться эта универсальность будет за счет ограничения возможностей начертания шрифта.

Кегль (font-size)

Кегль - это, если говорить упрощенно, размер шрифта. Более подробное объяснение следует искать в специальной литературе. Нам достаточно знать, что CSS через параметр font-size позволяет управлять размером букв.

Размер шрифта можно задавать в типографских пунктах (pt, 0,35 мм) или пикселах (px). При установке кегля следует помнить, что font-size задает не высоту буквы, а размер "очка" под букву, который больше самой буквы.

Вот несколько примеров использования font-size:([открыть](#))

```
<P >  
Кегль параграфа установлен в 12 пунктов</P>  
<P >  
Кегль параграфа установлен в 12 пикселов</P>  
<P >  
Кегль параграфа установлен в 120% от размера  
букв охватывающего параграф элемента</P>
```

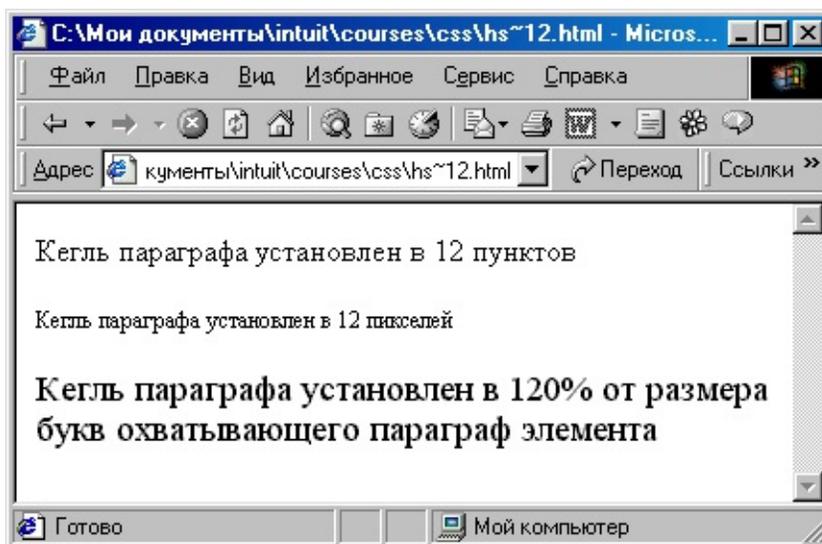


Рис. 10.7. Установка размера кегля в абсолютных и относительных единицах измерения.

Как видно из последнего примера, кегль можно задавать не только в абсолютных единицах, но и в относительных. Кроме процентов, существует еще несколько условных единиц измерения кегля, которые можно применять в CSS:([открыть](#))

```
<P >
```

Размер кегля large</P>
<P >
Размер кегля small</P>
<P >
Размер кегля x-small</P>
<P >
Размер кегля xx-small</P>

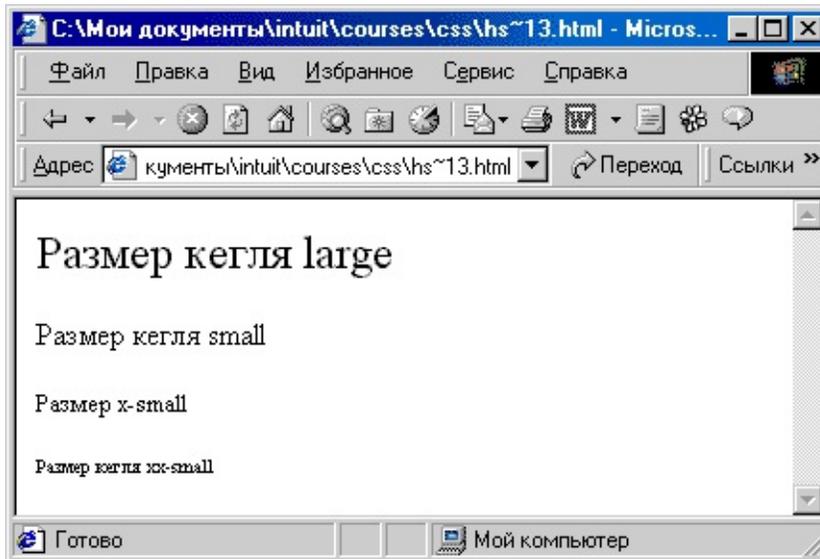


Рис. 10.8. Установка размера кегля в условных единицах измерения.

Аналогично x-small и xx-small, существуют размеры x-large и xx-large. Кроме того, есть larger, smaller и medium.

Начертание

У каждой гарнитуры (font-family) имеется несколько начертаний. Каждое из них определяется в CSS тремя параметрами стиля: font-style, font-variant, font-weight.

Атрибут стиля font-style определяет прямое начертание (normal) и курсив: [\(открыть\)](#)

<P >
Прямое начертание</P>
<P >
Курсив</P>

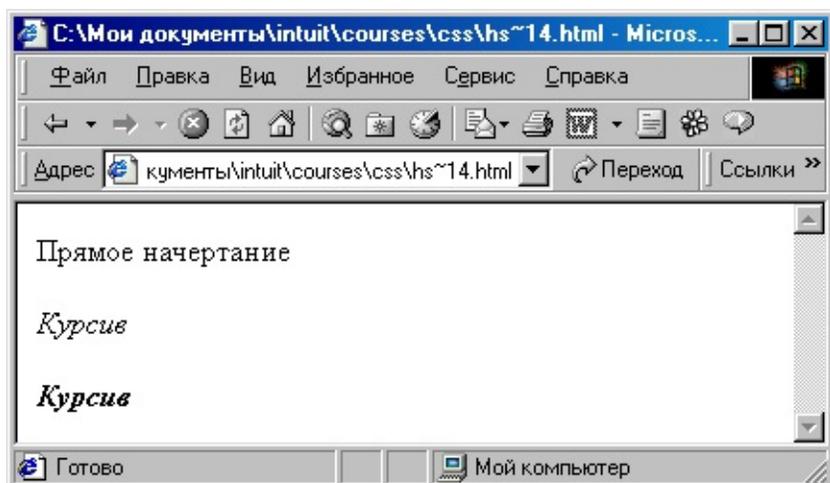


Рис. 10.9. Начертание гарнитур шрифтов.

Если хочется усилить насыщенность ("жирность") шрифта, то в описании стиля указывают атрибут `font-weight`, который принимает значения `normal` или `bold`:

```
<P >Курсив</P>
```

Хотя шрифт и масштабируется при помощи изменения кегля, качество его начертания при этом обычно страдает. Для качественного отображения мелких букв в некоторых гарнитурах присутствует начертание капители. В CSS для использования капители зарезервирован атрибут `font-variant`, который принимает значения `normal` и `small-caps`. На практике применение `font-variant` проблематично из-за отсутствия капители в стандартном наборе кириллических шрифтов.

4: Текст и списки

Текст

В этом разделе мы рассмотрим те свойства текстового фрагмента, которые остались без внимания в разделах, посвященных блокам текста и шрифтам.

При обсуждении свойств блочных элементов разметки речь шла о параметрах, относящихся к блоку как целому. Мы не рассматривали внутренние характеристики текста.

Рассказывая о шрифтах, мы акцентировали внимание на начертаниях символов как таковых, а не на их соотношении.

Тем не менее в стороне остались такие важные характеристики текстового фрагмента, как:

- межбуквенные расстояния;
- высота строк;
- выравнивание;
- отступ в первой строке параграфа;
- преобразования начертания.

Все эти атрибуты сгруппированы в свойства текстовых фрагментов (Text Properties).

Межбуквенные расстояния

Расстояние между буквами автоматически регулируется размером шрифта — кеглем. Чем больше размер шрифта, тем больше расстояние между буквами: ([открыть](#))

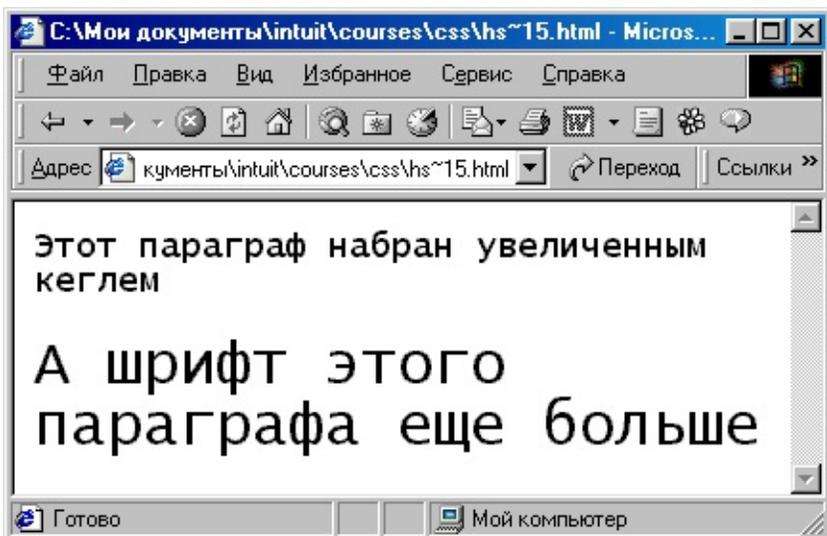


Рис. 11.1.

Присмотревшись внимательно, нетрудно убедиться, что расстояние между буквами в слове "параграф" первого примера и буквами слова "параграфа" второго примера разное. Во втором случае оно больше:



Рис. 11.2.

Моноширинный шрифт выбран не случайно. На пропорциональном шрифте межбуквенное расстояние зависит от начертания букв и показать его как расстояние между буквами достаточно сложно. У моноширинного шрифта размер символа фиксирован, поэтому и расстояние между буквами прослеживается четко.

Однако не всегда удобно управлять межбуквенным расстоянием через кегль (font-size). Бывают случаи, когда нужно либо уплотнить строку, либо увеличить расстояния между буквами. Это можно сделать с помощью атрибута letter-spacing: ([открыть](#))

```
Межбуквенное расстояние 5pt</P>  
<P STYLE=  
"font-family:monospace;  
letter-spacing:10pt;  
color:black">  
Межбуквенное расстояние 10pt  
</P>
```

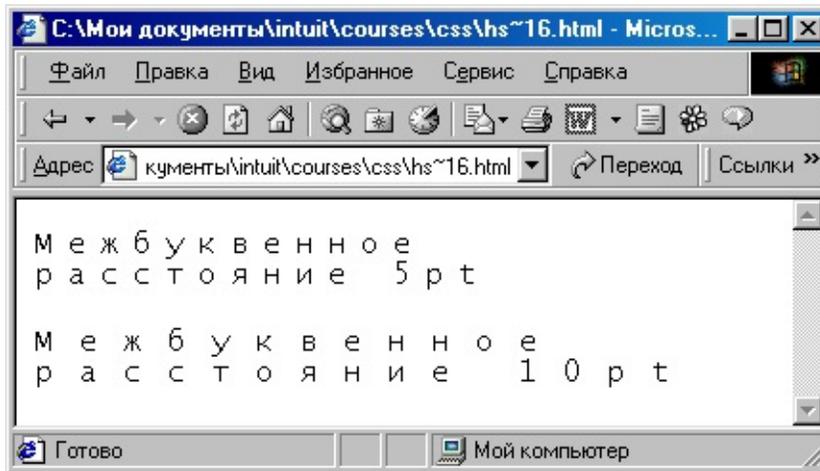


Рис. 11.3.

Правда, в версиях Netscape Navigator 4.x этот параметр не поддерживается.

Выравнивание

По умолчанию все слова в параграфе прижаты влево. Левый край параграфа таким образом оказывается выровненным. Точно так же может быть выровнен правый край параграфа или блока текста, и даже оба края вместе.

В обычной HTML-разметке такой эффект достигается за счет применения атрибута ALIGN, как это сделано на страницах данного пособия:

```
<P ALIGN=justify>...</P>
```

Аналогичный результат в CSS достигается за счет атрибута text-align: [\(открыть\)](#)

```
<P >
```

Этот параграф выровнен по правому краю. Все

строки справа кончаются на границе раздела.
А вот слева они начинаются с различным
отступом от левого края.</P>

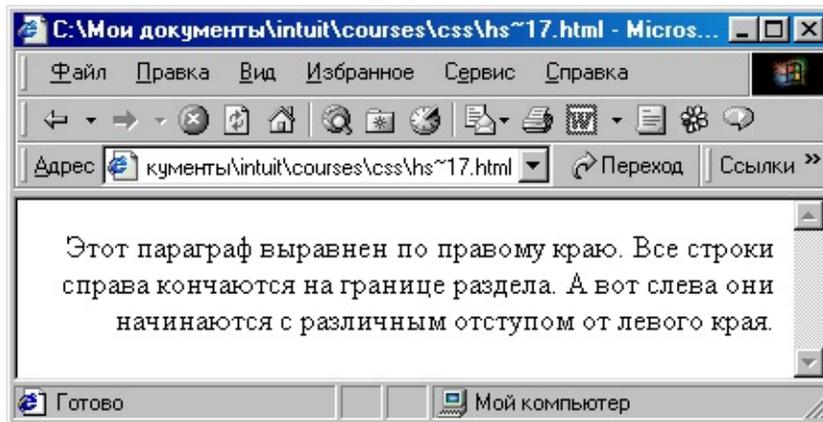


Рис. 11.4.

<P >

Этот параграф выровнен по левому и правому краям.
Все строки справа кончаются на вертикальной
границе раздела. Все строки слева теперь
начинаются также с вертикальной границы
раздела.</P>

(открыть)

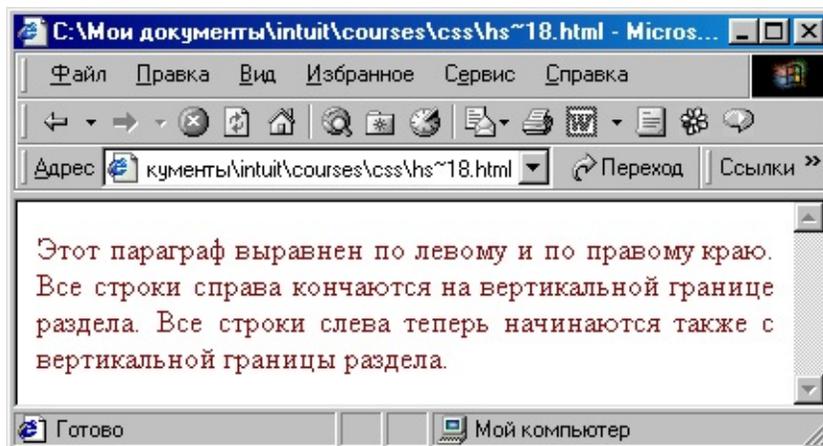


Рис. 11.5.

Выравнивать текст можно в любом блочном элементе. Причем можно не только выравнивать текст по краям блочного элемента, но и центрировать

его (<P >...</P>).

Преобразование шрифта

Преобразование шрифта подразумевает капитализацию слов, перевод всех "больших" и "маленьких" букв в большие, или, наоборот, получение одних строчных.

Рассмотрим несколько примеров: ([открыть](#))

<P >

Сделать заглавными</P>

<P >

Сделать строчными</P>

<P >

Сделать заглавными первые буквы в словах</P>

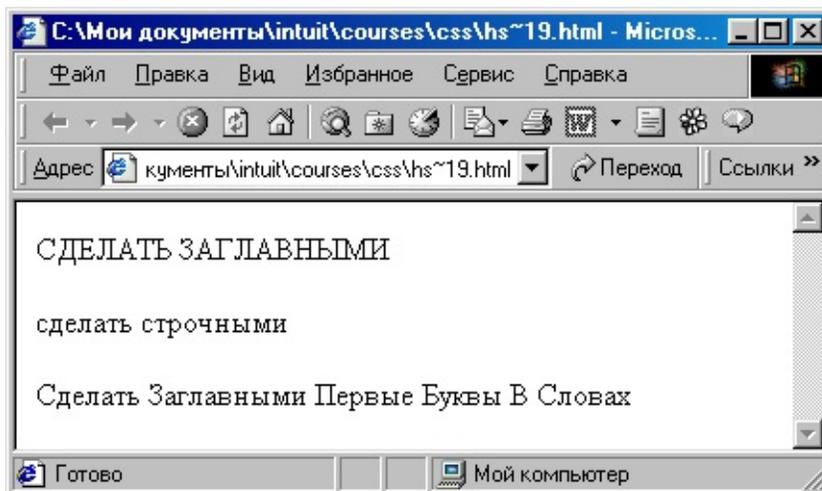


Рис. 11.6.

Обратите внимание, что выполнение преобразований зависит от алгоритма преобразования символов. В нелокализованных программах переход от строчных букв к прописным осуществляется путем простого смещения по таблице ASCII, что для русского алфавита не приемлемо.

Еще один вид преобразования шрифта — это подчеркивание, перечеркивание или надчеркивание слов. Выполняется такое преобразование с помощью атрибута text-decoration: ([открыть](#))

<P >
Перечеркнем это предложение.</P>
<P >
Подчеркнем это предложение.</P>

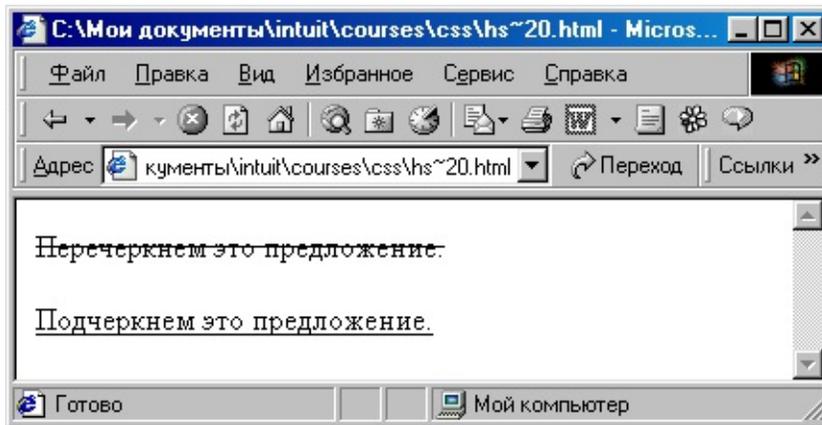


Рис. 11.7.

Для того, чтобы преобразование работало, необходимо соответствующее начертание (подчеркнутые или перечеркнутые начертания букв). Очень сложно найти гарнитуру, в которой было бы начертание с надчеркнутыми буквами. Отмена декора происходит, если использовать в text-decoration значение none.

Первая строка параграфа

При оформлении параграфов в технологии CSS автор может воспользоваться "красной" строкой, такую возможность предоставляет ему атрибут text-indent.

Речь идет о горизонтальном отступе в первой строке параграфа относительно его левого края: ([открыть](#))

<P >
Этот параграф мы начнем со строки с
горизонтальным отступом в двадцать
типографских пунктов от левого края параграфа.
</P>
<P >
А в этом параграфе мы применим отрицательный
горизонтальный отступ в первой строке

параграфа. </P>

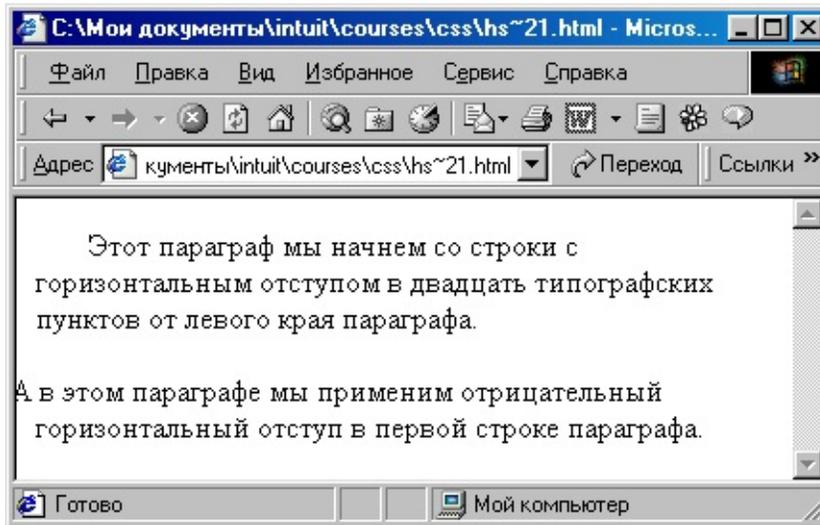


Рис. 11.8.

Отрицательные значения атрибутов — это нормальная практика CSS. Там, где применение отрицательного значения оправдано, например, в случае смещения вложенного блока текста относительно охватывающего элемента разметки, можно указывать отрицательные атрибуты смещения.

Кроме `text-indent` в CSS для оформления первой строки параграфа зарезервирован модификатор стиля `first-line`. Он позволяет не только задать горизонтальное смещение, но и определить другие параметры параграфа:

```
P:first-line { color:red; }
```

Еще один параметр, который влияет на отображение первой строки параграфа — первая буква первой строки. Ее отображением управляет модификатор `first-letter`:

```
P:first-letter { font-size:20pt; }
```

К сожалению, оба названных модификатора реализованы не во всех версиях браузеров, поэтому для верности применяют элементы разметки `FONT` и `TABLE`.

Межстрочное расстояние

В CSS межстрочное расстояние определяется параметром `line-height`. Он задает расстояние не между строками, а между базовыми линиями строк. Проще говоря, если, например, взять букву "н" и напечатать ее последовательно друг под другом, то `line-height` будет равно расстоянию между двумя одинаковыми точками букв.

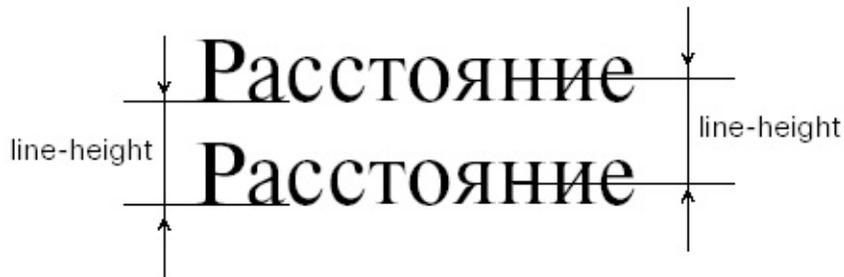


Рис. 11.9.

Посмотрим, как этот параметр влияет на взаимное расположение строк:
([открыть](#))

```
<P >
Этот параграф мы набрали кеглем 12 pt.
Line-height задан в 12 pt.</p>
<P >
Этот параграф мы набрали кеглем 12 pt.
Line-height задан в 24 pt.</P>
<P >
Этот параграф мы набрали кеглем 12 pt.
Line-height задан в 6 pt.</P>
```

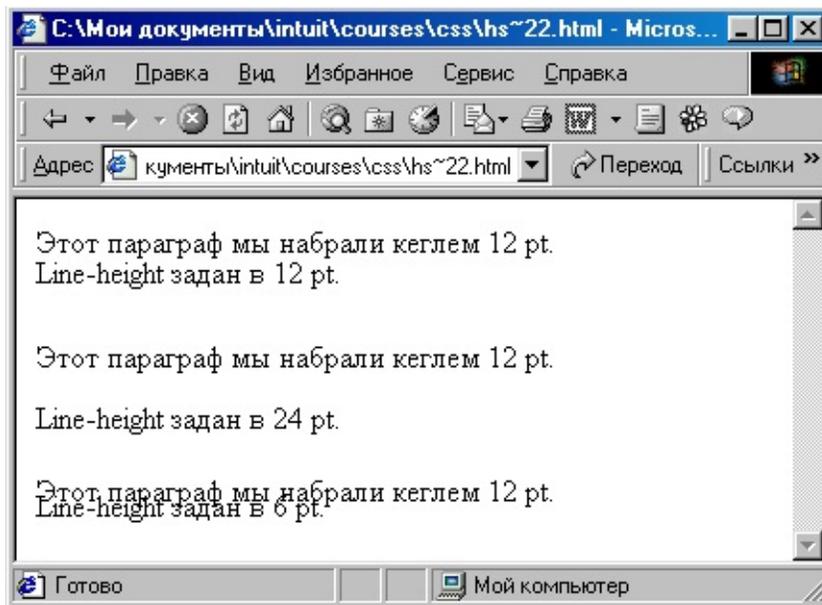


Рис. 11.10.

Первый пример набран со значением `line-height`, равным размеру кегля. Во втором примере значение `line-height` вдвое превышает размер кегля. В третьем примере значение `line-height` в два раза меньше размера кегля — строки стали "наползать" друг на друга.

В связи с использованием `line-height` следует обратить внимание на применение `in-line` картинок на HTML-страницах. Под `in-line` картинкой здесь имеется в виду картинка, которая встроена в тело документа при помощи элемента `IMG`, но не с новой строки и не как элемент таблицы: [\(открыть\)](#)

```
<P >
```

```
В эту строку мы встраиваем картинку - <IMG  
SRC="inline.gif" BORDER="0" WIDTH="24"  
HEIGHT="24" ALIGN="top">,  
на которой изображены концентрические круги.  
</P>
```

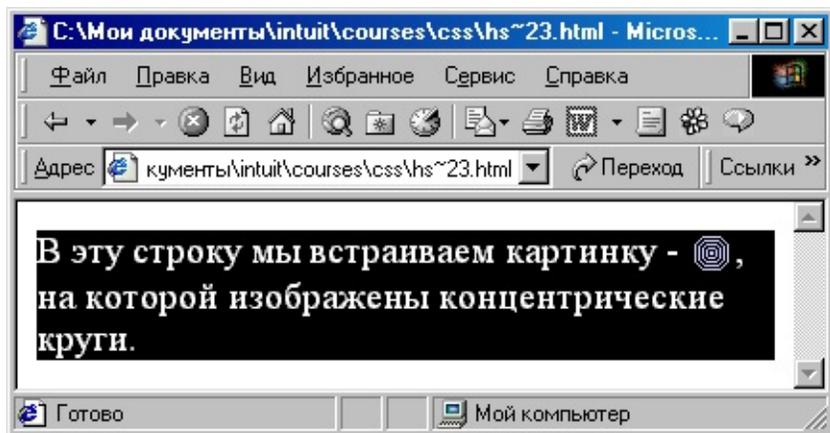


Рис. 11.11.

Картинка имеет размеры 24x24 пиксела и выравнена по верхнему краю строки. Ее размер больше размера кегля (20 px), поэтому межстрочное расстояние увеличено браузером автоматически. ([открыть](#))

<P >

В эту строку, которая имеет размер кегля 24px, мы встраиваем картинку - , на которой изображены концентрические круги.
</P>

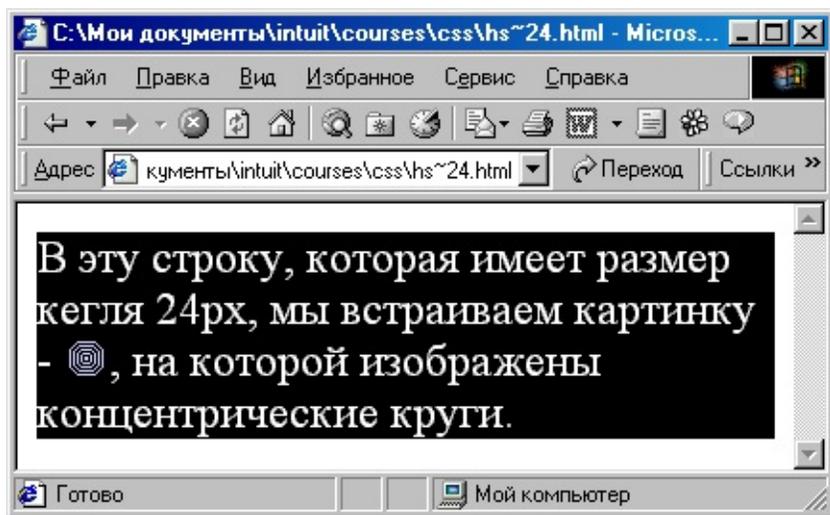


Рис. 11.12.

Таким образом, можно точно позиционировать текст и графику в строке.

Списки

При отображении списков CSS позволяет управлять формой и изображением "пулек" (bullets) списка. "Пулька" (bullet) — это символ, стоящий перед элементом списка. Например, в неупорядоченном списке (unordered list) перед элементом списка ставится "жирная" точка:

- Первый элемент списка
- Второй элемент списка
- Третий элемент списка

CSS позволяют управлять формой "пулек" и заменять "пульки" картинками.

Любопытно, что управление отображением элементов списка отнесено к набору свойств, в который входит атрибут `display`. У этого атрибута может быть только одно значение — `none`. Если элемент в своем описании имеет атрибут `display`, и этот атрибут равен `none`, то он не отображается браузером вообще:

```
<UL >  
<LI>Первый элемент списка  
<LI>Второй элемент списка  
<LI>Третий элемент списка  
</UL>
```

Если посмотреть HTML-код данного документа, то за примером описания списка следует код, который браузер не отобразил.

Атрибут `display` управляет отображением документа на дисплее компьютера, но не распространяется на другие среды отображения документа. Например, при печати скрытый список должен быть отображен.

Однако, на самом деле он не отображается и при печати.

Форма "пулек"

Форма "пульки" в виде "жирной" точки несколько непривычна. Обычно в машинописных документах используют черту. С другой стороны, в рекламных материалах часто в качестве "пульки" применяют квадрат или

другой символ типографского набора, а также графическую картинку.

CSS позволяет управлять формой "пульки" через атрибут `list-style-type`:
([открыть](#))

```
<UL >
<LI>В виде "пульки" используем квадрат
</UL>
<UL >
<LI>В виде "пульки" используем диск
</UL> <UL >
<LI>В виде "пульки" используем круг
</UL>
```

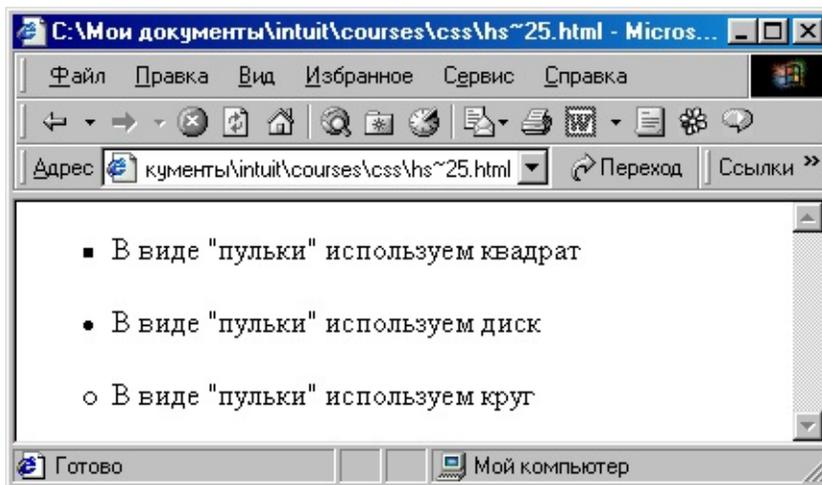


Рис. 11.13.

До сих пор мы обсуждали только неупорядоченные списки (UL), но управлять отображением "пулек" можно и в упорядоченных списках (OL):
([открыть](#))

```
<OL >
<LI>...
...
</OL>
<OL >
<LI>...
...
</OL>
<OL >
<LI>...
...
```

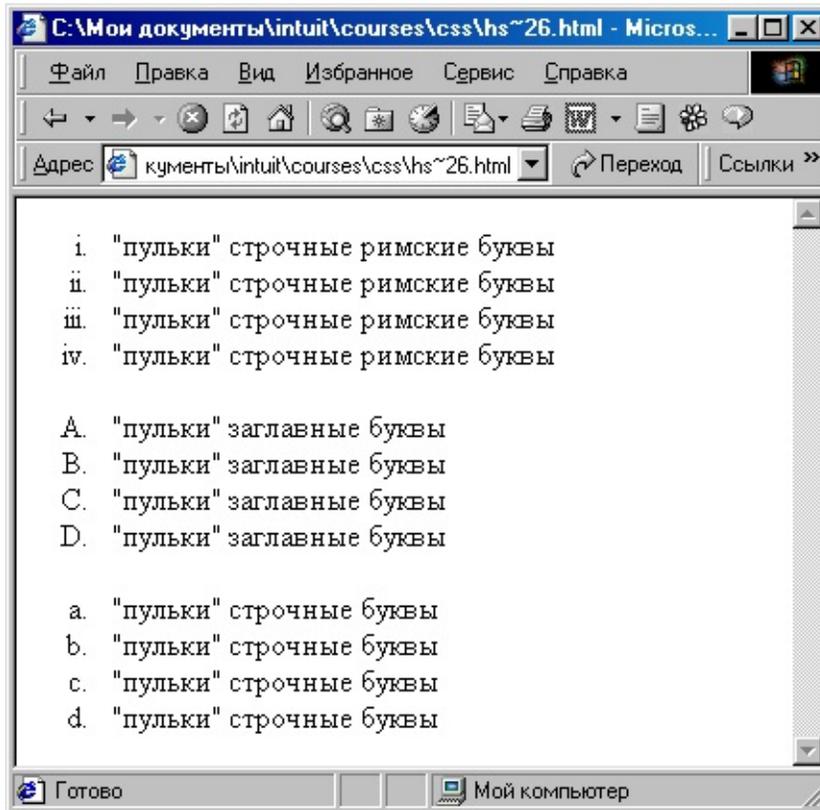



Рис. 11.14.

CSS позволяют вообще отказаться от "пулек". Для этого нужно указать значение атрибута `list-style-type` равным `none`.

"Пульки"-картинки

Если стандартные формы "пулек" автора страницы не устраивают, он может использовать нестандартные. Для этого ему придется "пульку" нарисовать самому и в виде графического файла разместить на Web-узле. У такой "пульки" есть URL, который используется в CSS для обращения к ней.

```
<UL  
> <LI>Элемент списка расположен за чертой  
</UL>
```

Картинка может быть и более замысловатой. Это уже зависит от фантазии автора документа. Например, можно создать картинку-стрелочку: ([открыть](#))

```
<UL>  
> <LI>Элемент списка расположен за стрелкой  
</UL>
```

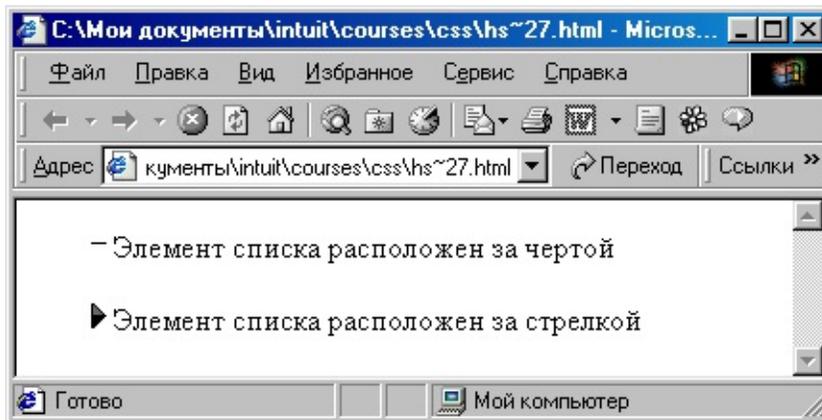


Рис. 11.15.

Здесь надо признаться в маленьком обмане. Если вы пользователь Internet Explorer, то все, что здесь написано — верно. Фрагмент кода, представленный перед примером, является его точной копией. Однако перед пользователями Netscape Navigator придется извиниться: Netscape Navigator этот атрибут не поддерживает. В тексте страницы присутствует JavaScript-код, который имитирует применение атрибута `list-style-image` в случае просмотра страницы браузером от Netscape.

5: Позиционирование

Этот раздел посвящен размещению блочных элементов HTML-разметки в рабочей области браузера с точностью до пиксела. До появления спецификации CSS-P, которая вошла в спецификацию CSS2, ничего подобного делать на HTML-страницах было нельзя. Некоторые наиболее пристрастные читатели могут спросить: "А как насчет элемента разметки LAYER?". Во-первых, LAYER был введен Netscape практически одновременно с появлением CSS-P, а во-вторых, он поддерживается только браузерами Netscape.

Позиционирование и его программирование на JavaScript - это сплошное "минное поле" между двумя основными браузерами. При просмотре этих страниц следует отдавать себе отчет в том, что для каждого браузера загружается своя страница описания свойств позиционирования и программирования этих свойств.

До появления CSS-P единственным средством относительно точного позиционирования были таблицы. Они позволяли точно расположить компоненты HTML-страницы относительно друг друга на плоскости. CSS-P позволяет точно разместить элемент разметки не только относительно других компонентов страницы, но и относительно границ страницы.

Кроме того, CSS-P добавляет странице еще одно измерение - элементы разметки могут "наезжать" друг на друга.

При этом можно менять порядок "наезда" - перекладывать слои. Чтобы в этом убедиться, достаточно воспользоваться ссылкой из приведенного примера.

Но и это еще не все.

Слои можно проявлять. ([открыть](#))

Рис. 12.1. Позиционирование

Рис. 12.2. Позиционирование

Термин "слой" вместо "блочный элемент разметки" используется здесь по той причине, что он лучше отражает эффект, который достигается за счет позиционирования, а вовсе не в пику приверженцам Microsoft.

Теперь переходим к обсуждению атрибутов позиционирования. ([открыть](#))

Рис. 12.3. Позиционирование

Рис. 12.4. Позиционирование

Координаты и размеры

Стандарт CSS-Р позволяет с точностью до пиксела разместить блочный элемент разметки в рабочем поле окна браузера. При таком подходе возникает естественный вопрос: как устроена система координат, в которой автор страницы производит размещение ее компонентов.

CSS-Р определяет две системы координат: относительную и абсолютную. Это позволяет обеспечить гибкость размещения элементов как относительно границ рабочего поля окна браузера, так и относительно друг друга.

Блоки - это не абстрактные точки, которые не занимают на плоскости страницы места. Блоки представляют собой прямоугольники, которые "заматают" площадь. Текст и другие компоненты страницы под блоком становятся недоступны пользователю, поэтому линейные размеры блока имеют для создания HTML-страниц не меньшее значение, чем его координаты.

Абсолютные координаты

При использовании "абсолютных" координат точка отсчета помещается в верхний левый угол окна браузера, а оси X и Y направлены вправо по горизонтали и вниз по вертикали, соответственно:

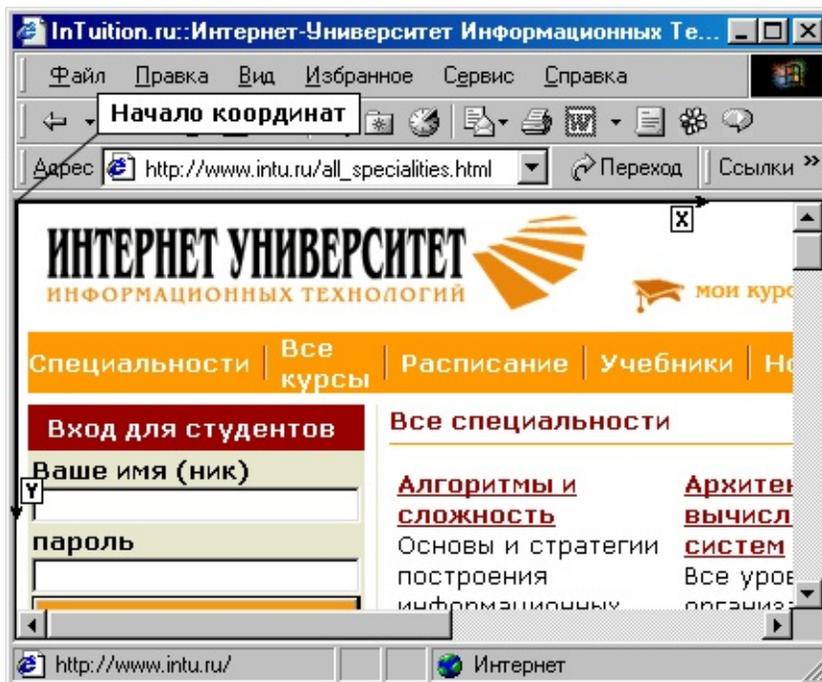


Рис. 12.5. Абсолютные координаты

Если в этой системе координат некоторый блочный элемент должен быть размещен на 10 px ниже верхнего обреза рабочей области браузера и на 20 px правее левого края рабочей области браузера, то его описание будет выглядеть следующим образом:

```
.example { position:absolute;top:10px;
           left:20px; }
```

В данной записи тип системы координат задан атрибутом position (значение - absolute), координата X задана атрибутом left (значение - 20 px), координата Y - атрибутом top (значение - 10 px).

Атрибуты top и left определяют координаты верхнего левого угла блока в абсолютной системе координат. ([открыть](#))

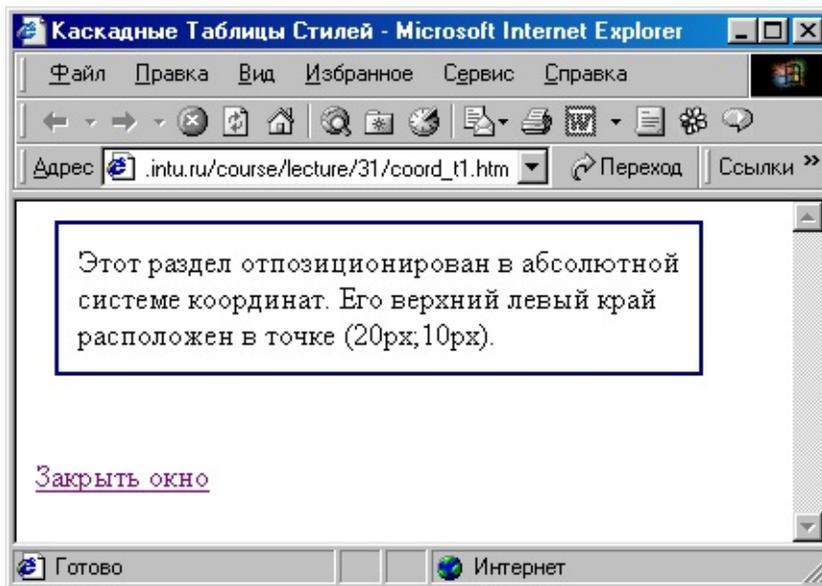


Рис. 12.6. Абсолютные координаты

Значения координат могут быть и отрицательными. Для того, чтобы убрать из отображаемой области блок с линейными размерами 100 px (высота) на 200 px (ширина), достаточно позиционировать его следующим образом: ([открыть](#))

```
.example { position:absolute;
            top:-100px;left:-200px;
            width:200px;height:100px; }
```

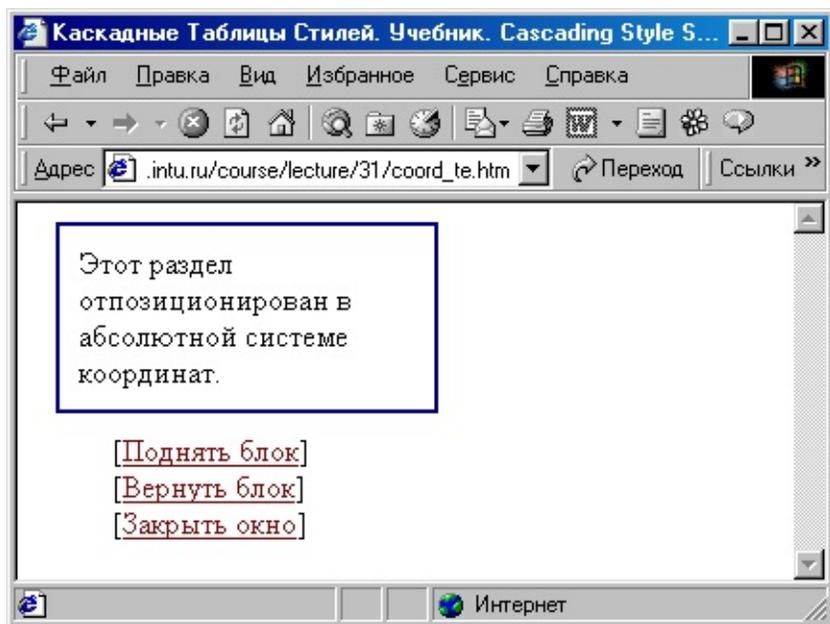


Рис. 12.7. Абсолютные координаты

Абсолютное позиционирование применяется тогда, когда либо все содержание страницы должно быть доступно без скроллинга ("прокрутки"), либо когда элементы разметки находятся в начале страницы и их взаимное расположение важно с точки зрения дизайна, например, для использования всплывающих меню.

Относительные координаты

Данная координатная система позволяет разместить блоки на странице в координатах охватывающего их блока. Преимущества такой системы координат очевидны: она позволяет сохранять взаимное расположение элементов разметки при любом размере окна браузера и его настройках по умолчанию.

В качестве точки отсчета в этой системе координат выбрана точка размещения текущего блока по умолчанию. Ось X при этом направлена горизонтально вправо, а ось Y - вертикально вниз.

Чтобы задать координаты блока, в этой системе применяют запись типа: [\(открыть\)](#)

<DIV >

```
<DIV >
Этот блок находится в точке отсчета
относительных координат
</DIV>
<DIV >
А этот блок смещен вправо на 50px
</DIV>
</DIV>
```

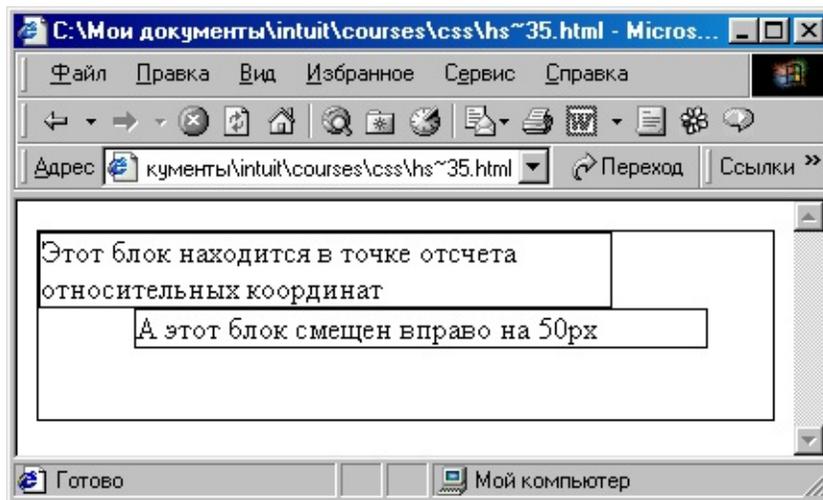


Рис. 12.8. Относительные координаты

Для работы с относительной системой координат лучше пользоваться универсальными блоками DIV. Это связано с тем, что в Netscape Navigator, например, параграф не может содержать параграфов. Любой блок немедленно закрывает параграф, следовательно, вложить в него что-либо нельзя.

Следует отметить, что Netscape Navigator вообще непредсказуем в работе с относительными координатами, поэтому в нем следует их избегать.

В относительной системе координат можно пользоваться отрицательными смещениями: ([открыть](#))

```
<DIV >
<A HREF="javascript:
if(flag==0)
{ window.document.layers[2].left=-50;flag=1; }
else
{ window.document.layers[2].left=50;flag=0; };
void(0);">
```

Сдвинуть слой

</DIV>

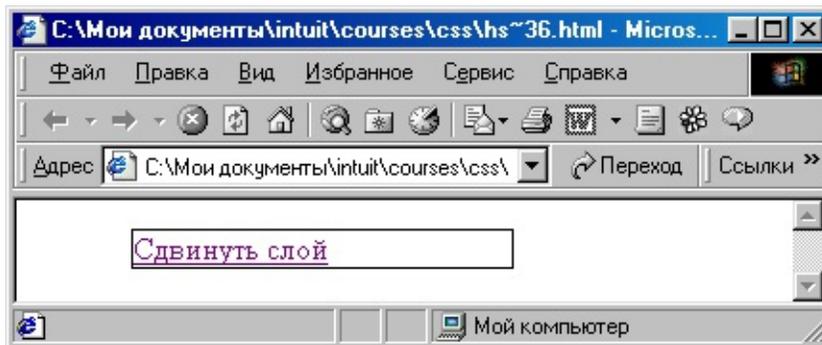


Рис. 12.9. Относительные координаты

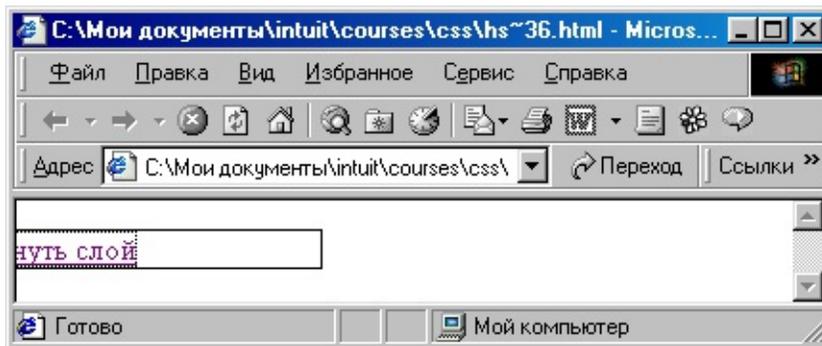


Рис. 12.10. Относительные координаты

В данном примере слой, первоначально сдвинутый на 50 пикселей вправо, после нажатия на гипертекстовую ссылку смещается на 100 пикселей влево, получая отрицательную величину смещения по оси X (`left:-50 px`). После повторного нажатия на ссылку положение блока восстанавливается.

Линейные размеры блока

Линейные размеры блока задаются двумя параметрами: шириной (`width`) и высотой (`height`) блока. В браузерах ширина и высота блока интерпретируется по-разному.

В Netscape Navigator ширина и высота блока - рекомендуемые параметры. Если текст выходит за эти ограничения, то блок увеличивается до необходимых размеров, а если текста нет вообще, то блок сжимается до

маленького квадрата:

```
<P >  
...  
</P>
```

Приведенного в примере описания достаточно для получения результата, но в Netscape Navigator для такого блока нужно применить некоторые дополнительные атрибуты: ([открыть](#))

```
<P >  
<SPAN >  
...  
</SPAN>  
</P>
```

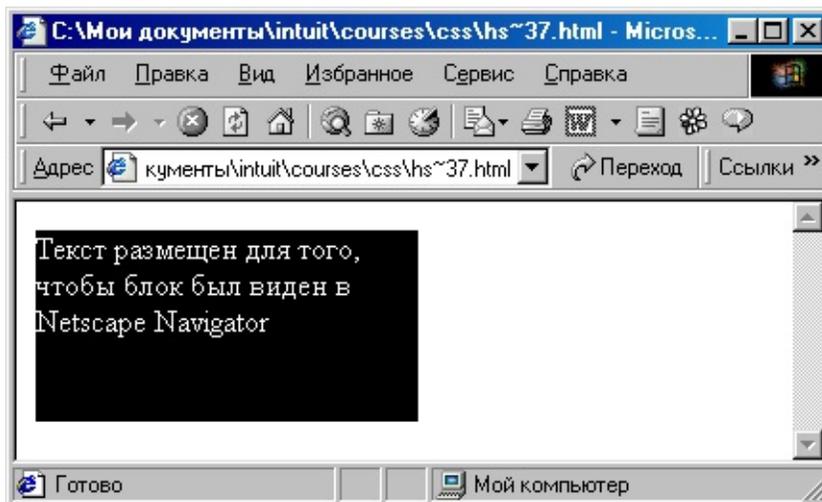


Рис. 12.11. Линейные размеры блока

Без границы блок не будет залит черным цветом, а без span текст будет отображаться цветом данной страницы по умолчанию. Никакому разумному объяснению такое поведение браузера не поддается, поэтому строить дизайн страниц на этих атрибутах не стоит.

Управление видимостью

Одним из наиболее интересных атрибутов CSS-P является visibility. Он позволяет "проявлять" или прятать информацию на HTML-странице.

Например, страница содержит набор кадров, которые можно

перелистывать. Для этого просто нужно нажимать на слово "далее" в конце страницы.

Перелистывание реализовано как изменение атрибута `visibility` в JavaScript-функции:

```
function change()
{
next=current+1; if(next>1) next=0;
window.document.all.item("list",
    current).visibility="hidden";
window.document.all.item("list",
    next).visibility="visible";
current=next; }
```

Последовательность "list" - это значение атрибута ID раздела.

Вызов функции при этом помещен в гипертекстовую ссылку со схемой URL "javascript":

```
<A HREF="javascript:change();void(0);">
далее...
</A>
```

На примере предыдущих фрагментов страницы мы рассмотрели программирование `visibility`, но ни словом не упомянули о том, как данный атрибут записывается в CSS. Здесь мы восполняем этот пробел и приводим описание невидимого раздела:

```
<DIV ID=test >...</DIV>
```

При программировании атрибута `visibility` следует принимать в расчет тип браузера. Данная страница написана для Microsoft Internet Explorer. Это следует и из программного кода, который поддерживает объектную модель документа DHTML, и из использования атрибута ID у раздела.

Порядок наложения и область видимости

Абсолютное и относительное позиционирование блоков позволяет придать странице третье измерение. Координаты блока, а точнее, левого верхнего угла прямоугольника блока, и его линейные размеры определяют местоположение блока на плоскости. Поскольку блок не является

абстрактной точкой, прямоугольники блоков перекрываются. Причем в определенном порядке, что позволяет говорить об их размещении в пространстве. Порядок наложения (перекрытия) блоков определяется атрибутом z-index.

Вообще, при позиционировании блоков и их наложении друг на друга возникают видимые и невидимые области блоков. CSS позволяет управлять видимостью прямоугольной части блока. За это в спецификации CSS-P отвечает атрибут clip.

Порядок наложения блоков. z-index

Если быть более точным, то блоки - это прямоугольные карточки, которые лежат на поверхности, перекрывая друг друга. Расстояние между ними задать нельзя, т.е. полноценного третьего измерения нет:

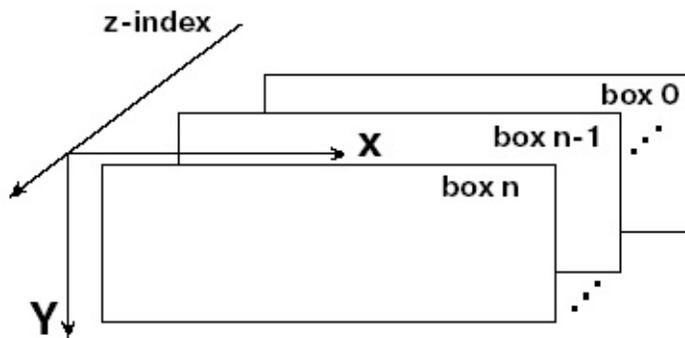


Рис. 12.12. Порядок наложения блоков. z-index

Порядок перекрытия блоков (слоев, в терминах Netscape) определяется атрибутом z-index. Чем больше значение z-index, тем ближе к наблюдателю находится слой:

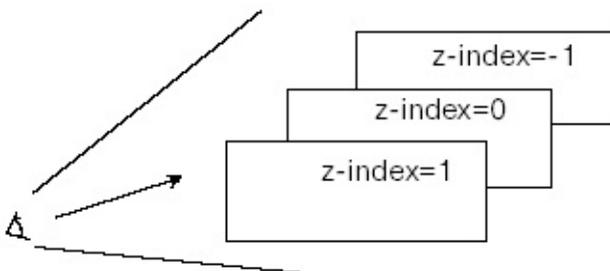


Рис. 12.13. Порядок наложения блоков. z-index

Как мы видим, z-index может принимать и отрицательные значения.

При работе с этим атрибутом следует иметь в виду, что Netscape Navigator и Internet Explorer применяют разные модели описания "слойки". В Internet Explorer это просто числовой параметр, который закреплен за блоком и не влияет на значения данного параметра у других блоков. В Netscape Navigator изменение параметра z-index одного блока влияет на значения этого параметра у других блоков. Они, словно карточки на столе, переключаются и получают новые номера.

Карточки перекрываются. Если нажать на гипертекстовую ссылку "Переложить", то голубая карточка поползет вниз и при обратном ходе ее положение относительно красной карточки изменится. ([открыть](#))

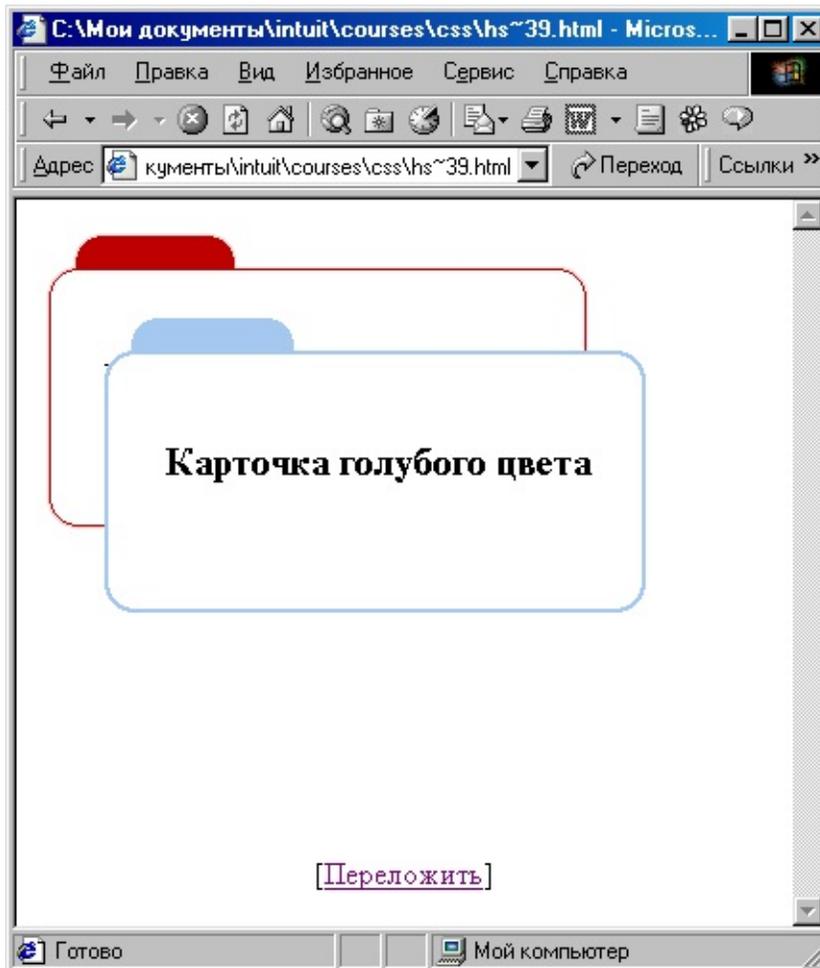


Рис. 12.14. Порядок наложения блоков. z-index

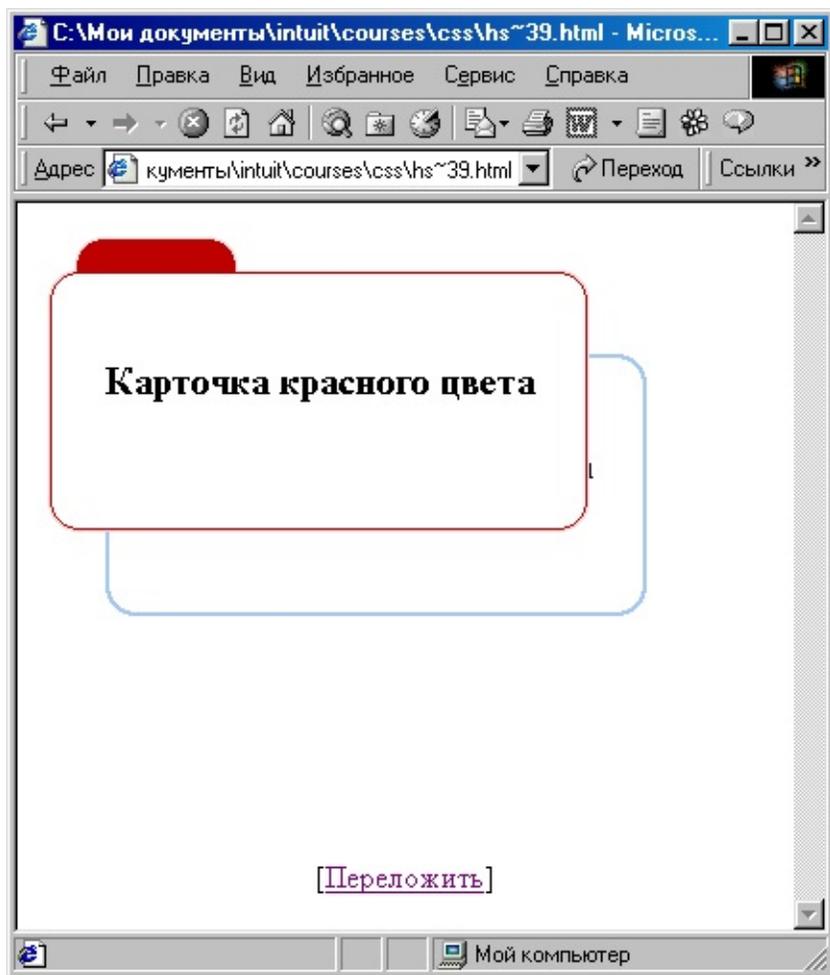


Рис. 12.15. Порядок наложения блоков. z-index

Область видимости блока. clip

Совершенно необязательно показывать весь блок целиком. Можно показать только часть блока. Управляется такое отображение параметром clip описания CSS:

```
<DIV >  
...  
</DIV>
```

При вырезании области видимости в Internet Explorer следует учитывать, что работает clip только при абсолютном позиционировании. В Netscape Navigator поддерживается как абсолютное позиционирование, так и относительное.

Литература

1. Храмцов П.Б., Брик С.А., Русак А.М., Сурин А.И.
Основы web-технологий
БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий - ИНТУИТ.ру, 2007
2. **Cascading Style Sheets, level 1 (CSS1) W3C Recommendation 17 Dec 1996, revised 11 Jan 1999.**
Рекомендация W3C для CSS1. Фактически, -это стандарт.
3. **Hakon Wium Lie и Bert Bos. Каскадные таблицы стилей, уровень 1**
Перевод Всероссийского Клуба Вебмастеров Перевод спецификации CSS1.
4. **Hakon Wium Lie, Bert Bos: Cascading Style Sheets: Designing for the Web 2nd edition, Addison-Wesley 1999, ISBN:0-201-59625-3 [CSS2].**
Примеры отдельных глав из книги.
5. **Каскадные таблицы стилей II. Спецификация CSS2**
Рекомендации W3C от 12 мая 1998 года. Частичный перевод спецификации CSS2. Переводчики обещают, что смогут в конце концов его закончить.
6. **Cascading Style Sheets, level 2. CSS2 Specification W3C Recommendation 12-May-1998.**
Рекомендация W3C для CSS1. Фактически, это стандарт.
7. **The CSS1 Test Suite**
Страница W3C для проверки браузеров на предмет поддержки ими спецификации CSS1.
8. **Erik Wilde: Wilde's WWW, technical foundations of the World Wide Web. Springer 1998,ISBN:3-540-64285-4 [CSS1].**
Книга, охватывающая все информационные технологии World Wide Web.
9. **Ian Graham: The HTML Stylesheet Sourcebook**
Текста книги нет, но есть работающие примеры применения CSS.
10. **Handan Selamoglu. A User's Guide to Style Sheets. Microsoft Corporation, Updated: October 24, 1996.**
Очень полезное, прекрасно структурированное и краткое руководство. К нему прилагается справочник по атрибутам CSS.
11. **Tapio Markulalle. CSS teaching pages**
Очень подробный учебник по CSS, написанный изначально на финском языке и переведенный автором на английский язык.

12. **Urban Fredriksson. Short Guide to CSS**

Одна страничка с коротким описанием принципов использования CSS.

13. **Style Sheets**

Подборка статей по использованию CSS.