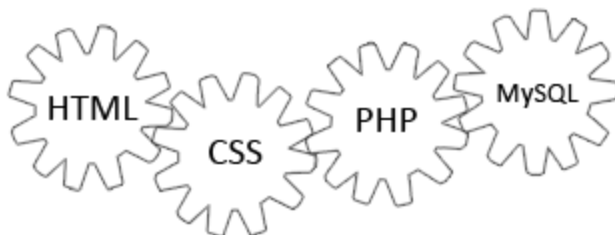

Московский городской Дворец детского (юношеского) творчества
отдел оборонно-массовой работы
сектор новых информационных технологий

Создание сайтов:
HTML, CSS, PHP, MySQL
(часть первая: лекции и практические задания)



Москва 2010 г.

УДК 004.438

Росс В. С. Создание сайтов: HTML, CSS, PHP, MySQL. Учебное пособие, ч. 1 — МГДД(Ю)Т, М.:2010 – 107 с.

Литературный редактор: Разуваева А. А.

Учебное пособие предназначено для подготовки учащихся, специализирующихся в области разработки веб-сайтов. Описываются современные технологии создания сайтов: язык разметки гипертекста HTML, каскадные таблицы стилей CSS, язык программирования PHP и СУБД MySQL. Пособие может быть использовано для организации занятий в рамках дополнительного детского образования, факультативных занятий в ВУЗе и т.п.

Первая часть пособия содержит теоретический материал и практические задания по языку HTML и технологии CSS.

Содержание

Тема 1. Основы Web-мастеринга	7
Лекция 1.1. Основы WWW	7
Тема 2. Язык HTML	12
Лекция 2.1. Основы HTML	12
Лекция 2.2. Основные теги, работа с текстом, списки	18
Лекция 2.3. Создание ссылок	23
Лекция 2.4. Изображения	25
Лекция 2.5. Создание таблиц	31
Лекция 2.6. Кодировки текста и специальные символы	37
Тема 3. Технология CSS	42
Лекция 3.1. Основы CSS	42
Лекция 3.2. CSS-свойства: размеры, цвета, шрифты, текст	50
Лекция 3.3. CSS-свойства: поля, заполнение, границы	56
Лекция 3.4. CSS-свойства: фон, оформление таблиц	64
Лекция 3.5. Теги DIV и SPAN, псевдоклассы	73
Лекция 3.6. CSS-свойства: позиционирование	81
Тема 4. Верстка сайтов.	93
Лекция 4.1. Основы верстки. Табличная верстка.	93
Лекция 4.2. Блочная верстка	103

Введение

Пособие посвящено актуальной теме – разработке сайтов.

В первой и второй части пособия содержатся конспекты лекций с примерами, ссылки на Интернет-источники, посвященные рассматриваемой теме, и практические задания двух уровней сложности. Задания повышенной сложности отмечены знаком (*). В третьей части приводятся решения практических заданий для всех изучаемых тем. Пособие охватывает основные средства разработки современных сайтов. Так как охватить все аспекты веб-технологий в рамках курса не представляется возможным, некоторые теги HTML, правила CSS, функции PHP и т.п. в пособии не рассматриваются.

Организация занятий

В рамках курса предполагается проведение лекционных и практических занятий в очной или дистанционной форме.

На лекционных занятиях педагог предлагает для изучения учебный материал, происходит первичное закрепление новых знаний. Для проведения очных лекций рекомендуется использовать компьютер и проектор для демонстрации примеров, приведенных в материалах лекций. Рекомендуется вносить в код примеров изменения и демонстрировать их влияние на отображение веб-страницы или результат работы программы.

После лекционного проводится практическое занятие, на котором учащиеся закрепляют либо осваивают материал на поставленных педагогом задачах. Сначала учащимся предлагаются задачи нормального уровня сложности. Учащиеся, успешно их выполнившие, решают задачи повышенного уровня. Нерешенные на занятии задачи могут быть заданы для выполнения дома. Если у учащегося возникают сложности с выполнением заданий, ему может быть предоставлен листинг решения или его часть из второго тома пособия.

Программное обеспечение

Для реализации задач из тем 2-10 необходим установленный на компьютере браузер и текстовый редактор. Для тестирования страниц со сложными CSS-стилями желательно просматривать их в различных браузерах, например в Internet Explorer 6-8, Mozilla Firefox и Opera. Особо тщательно следует по-

дойти к выбору текстового редактора. Использование обычного редактора замедляет скорость работы, ведет к увеличению количества ошибок в коде. Необходима специализированная программа для разработчиков с поддержкой подсветки и свертывания синтаксиса HTML, CSS и PHP в одном файле. Желательно также наличие автодополнения и справки по аргументам функций, инструмента выбора цвета и т.п. Этим требованиям отвечают такие бесплатные редакторы, как Notepad++ (<http://notepad-plus.sourceforge.net/ru/site.htm>), SciTE (<http://code.google.com/p/scite-ru/>) и другие.

Для решения задач из тем 5-10 необходимо организовать доступ учащихся к веб-серверу Apache и серверу СУБД MySQL. Для ОС семейства Unix наиболее простым способом является установка «пакетов» («портов»), для ОС семейства Windows – готовых сборок, таких как WampServer (<http://www.wampserver.com/en/>) или XAMPP (<http://www.apachefriends.org/>). Использование пакета Денвер не рекомендуется, т.к. он более сложен в использовании и настройке. Все вышеперечисленные продукты являются бесплатными.

Возможно несколько вариантов установки:

1. Установка на каждый компьютер в учебном классе. Наиболее простой сценарий, но и наименее гибкий. При использовании ОС Windows для запуска серверов обязательно наличие прав администратора.
2. Установка на выделенный сервер в локальной сети. В этом случае необходимо выделить для каждого учащегося отдельный каталог и базу данных на сервере, обеспечить удаленный доступ и разделение прав пользователей. Преимуществом данного способа является приближение к реальным условиям разработки, возможность совместной работы учащихся и выполнения задач с любого рабочего места.
3. Установка выделенного сервера, аналогичная предыдущему варианту, но с доступом через Интернет. Реализуется на круглосуточно работающем сервере с прямым подключением к глобальной сети (с выделением внешнего IP-адреса) или покупкой услуги хостинга.

Наиболее предпочтителен последний сценарий, т.к. при нем возможна удаленная работа учащихся из дома и упрощается презентация результатов на выставках, конференциях и т.п. Если такой способ является слишком затратным, то можно комбинировать первые два варианта: при наличии компьютера дома учащиеся устанавливают на него веб-сервер и СУБД, а в классе пользуются общим выделенным сервером.

Рекомендуется использовать последние стабильные версии Apache, MySQL и оболочки phpMyAdmin или готовых сборок (WampServer, XAMPP).

Эти программы не предъявляют серьезных требований к аппаратному обеспечению и могут быть запущены на относительно «слабых» компьютерах.

Условные обозначения, принятые в пособии



Ресурсы Интернета, посвященные материалу, изложенному в лекции. Материалы, помеченные звездочкой (*), являются дополнительными и содержат сведения, не упомянутые в лекции.



Практические задания. Задания, помеченные звездочкой (*), являются усложненными.



Важное примечание для учащегося.



Важная информация для педагога.



Дополнительная информация для учащегося. При чтении лекции может быть пропущена.

Шрифтом Courier выделяется программный код, а также ключевые слова (теги, атрибуты, свойства, значения, операторы, конструкции, функции и т.д.), HTML, CSS, PHP и SQL.

Замечания и предложения просьба отправлять автору на электронную почту: vladislav.ross@gmail.com.

Лекция 1.1. Основы WWW.



При чтении курса слабо подготовленной аудитории рекомендуется пропустить эту лекцию и начать обучение с основ HTML или изложить ее сокращенно.

Основные понятия

WWW (World Wide Web – «всемирная паутина») – глобальное информационное пространство, основанное на физической инфраструктуре Интернета и протоколе передачи данных HTTP.

Понятие WWW часто путают с понятием **Интернет** – глобальной телекоммуникационной сетью. Интернет состоит из огромного количества компьютеров и сетей, в то время как всемирную паутину составляет множество веб-сайтов. Помимо WWW посредством Интернета работает множество различных служб: e-mail, IP-телефония, Интернет-радио и телевидение, файловые серверы, компьютерные игры и др.

Название «Интернет» происходит от англ. *Interconnected Networks* – объединенные сети. Это объединение можно представить на таком примере: компьютеры в классе объединены в сеть, эта сеть является звеном в сети учреждения, сеть учреждения подключена к сети провайдера Интернет, провайдер – к более крупному городскому или международному провайдеру. Связь между континентами осуществляется через кабели, проложенные по дну океанов.

HTTP (Hypertext Transfer Protocol – «протокол передачи гипертекста») – предназначен для установления связи с веб-сервером и обеспечения доставки HTML-страниц веб-браузеру клиента. Иначе говоря, HTTP – это «язык», на котором общаются браузер и сервер.

Гипертекст – размеченный текст, содержащий в себе ссылки на внешние ресурсы.

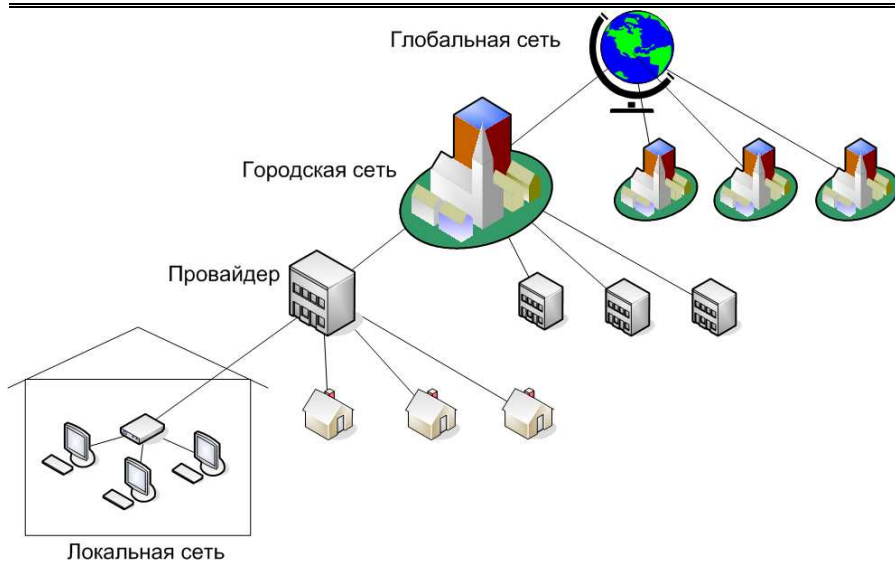


Рисунок 1.1. Объединение сетей в Интернет

Веб-страница – гипертекстовой ресурс Всемирной паутины, обычно написанный на языке HTML. Веб-страница может содержать ссылки для перехода на другие страницы, а также изображения, медиафайлы, например звуковые файлы и видео, Flash-анимацию и т.п.

Программа, демонстрирующая веб-страницу, называется **веб-браузер**. Несколько веб-страниц, объединенных общей темой и дизайном, образуют **веб-сайт**.

Веб-сервер (HTTP-сервер) – это программное обеспечение (ПО), предоставляющее доступ к сайтам. Наиболее популярными веб-серверами являются Apache (для ОС Windows и Unix) и Microsoft IIS (для Windows). Также веб-сервером называют компьютер, на котором установлено такое ПО.

URL (Uniform Resource Locator) – адрес ресурса.

Структура URL:

протокол :// пользователь : пароль @ хост : порт / путь ? запрос

Некоторые элементы URL могут быть опущены. Чаще всего адрес имеет вид протокол :// хост / путь ,

где протокол – http, https, ftp, skype и др.

хост – доменное имя (или IP-адрес)

путь – местонахождение ресурса на хосте

Например: <http://www.redut.ru/forum/index.php>

При помощи URL можно сослаться на любой открытый ресурс, будь то страница, изображение, файл для загрузки и т.д.

Для обращения к сайту используют **доменное имя**. Доменное имя состоит из нескольких доменов, разделенных точкой.

Например:

maps.google.com – означает, что домен третьего уровня maps входит в домен второго уровня google, который в свою очередь входит в домен первого (верхнего) уровня com.

Домены верхнего уровня делятся на общие и географические. Общие домены предназначены для определенного класса организаций, например:

.com – для коммерческих сайтов;

.org – для некоммерческих организаций;

.net – для сайтов, чья деятельность связана с Интернетом.

Географические домены выделяются для конкретной страны, например:

.ru – Россия;

.us – США

.eu – Европейский союз

.de – Германия

Использование географического домена не всегда означает то, что сайт размещается в соответствующей стране или имеет к ней какое-либо отношение. Например, многие телекомпании используют домен .tv островного государства Тувалу.

Помимо доменного имени для работы сайта необходим круглосуточно работающий веб-сервер, способный выдержать большие нагрузки. Услуга размещения веб-сайта на веб-сервере называется **хостингом**. Хостинг может быть платным или бесплатным. На платном хостинге, как правило, предоставляется доменное имя второго уровня. За пользование хостингом и доменом взи-

мается абонентская плата. На бесплатном хостинге предоставляется доменное имя третьего уровня (напр. `example.narod.ru`). По сравнению с платным хостингом, бесплатный имеет ограниченную функциональность, возможен принудительный показ рекламы на размещаемом сайте и т.п.

Механизм работы:

1. Браузер открывает соединение с сервером
2. Браузер отправляет серверу запрос на получение страницы
3. Сервер формирует ответ (HTML-код или, например, картинку) браузеру и закрывает соединение
4. Браузер обрабатывает HTML-код и отображает страницу

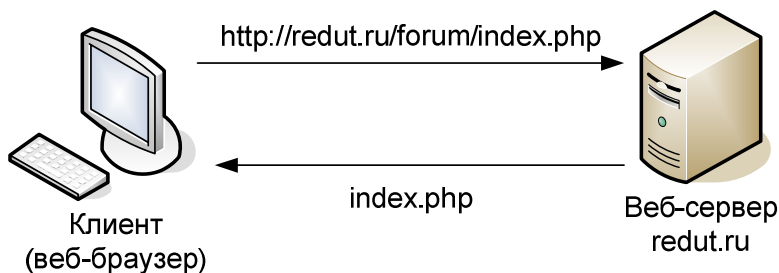


Рисунок 1.2. Механизм взаимодействия



Ресурсы в Интернете

- Интернет. <http://ru.wikipedia.org/wiki/Интернет>
- Всемирная паутина. <http://ru.wikipedia.org/wiki/Www>
- Доменное имя. http://ru.wikipedia.org/wiki/Доменное_имя
- Список национальных доменов верхнего уровня http://ru.wikipedia.org/wiki/Список_национальных_доменов_верхнего_уровня



Задания:



Задания в этой теме рекомендуется сначала выполнить вместе с преподавателем и затем повторить дома.

а) С помощью команды `tracert` (в Windows) или `tracertoute` (в Linux) определите маршрут передачи данных до удаленных отечественного и зарубежного узлов в Интернете (например, www.yandex.ru и www.google.com). Если работе команды препятствуют настройки брандмауэра, воспользуйтесь аналогичным сервисом в Интернете, например <http://ping.eu/>.

б) Используя службу Whois, получите регистрационные данные выбранных доменов.

в) Измерьте вашу скорость подключения к Интернету с помощью сервиса <http://speedtest.net> или аналогичного.

г*) В дополнении FireBug для браузера Mozilla или аналогичном инструменте изучите работу протокола HTTP: заголовки запроса браузера, заголовки ответа сервера, MIME-типы ресурсов.

Лекция 2.1. Основы HTML

Язык HTML (от англ. *HyperText Markup Language* – «язык разметки гипертекста») служит для создания веб-страниц. Большинство сайтов созданы именно с помощью HTML.

Синтаксис HTML

HTML-документы представляют собой файлы с текстом и дополнительными инструкциями языка HTML, называемыми **тегами**. Теги позволяют задавать форматирование текста, а также размещать в документе мультимедийные файлы (изображения, звук, Flash-анимацию), гипертекстовые ссылки на другие документы, табличные данные, формы ввода данных. HTML-документы имеют расширение имени файла htm или html. Редактирование HTML кода производят в текстовом редакторе (например, в обычном блокноте), а просмотр – в браузере.

Структура тега:

```
<имя тега атрибут1 атрибут2="значение2" ...>
```

Тег состоит из имени тега, за которым может следовать список атрибутов, помещаемых между открывающей и закрывающей угловыми скобками (< и >). Атрибуты позволяют управлять поведением тега. Они могут иметь конкретные значения, задаваемые после знака равенства. Значения атрибутов заключаются в одиночные или двойные кавычки ("). Атрибуты отделяются друг от друга пробелом, порядок следования атрибутов значения не имеет. Имена тэгов и атрибутов нечувствительны к регистру.

Пример: ``

Тег FONT предназначен для управления внешним видом текста. В примере он задает начертание текста шрифтом Arial, красным цветом.

Теги подразделяются на парные и непарные. Парные теги имеют закрывающий тег, непарные – не имеют. Закрывающий тег содержит косую черту пе-

ред именем и не имеет атрибутов. Между открывающим и закрывающим тегами помещается текст и другие теги. Атрибуты указываются только в открывающем теге.

Для выделения текста жирным используется тег ``. Пример:

HTML-код: текст ``жирный текст`` текст

В браузере: текст **жирный текст** текст

Примером непарного тега является тег `
` – перевод строки. Обычный перевод строки клавишей {Enter} браузер игнорирует (как и несколько поставленных подряд пробелов или знаков табуляции).

Неправильно:

HTML-код:

первая строка

вторая строка

В браузере:

первая строка
вторая строка

Правильно:

HTML-код:

первая строка`
`вторая строка

В браузере:

первая строка

вторая строка

Парный тег обязательно должен иметь закрывающий! Например, если не закрыть тег ``, весь текст на странице за ним станет жирным.

При вложении тегов друг в друга закрывать теги нужно начиная с самого последнего, в обратном порядке.

Тег `<i>` используется для выделения текста курсивом.

Неправильно: HTML-код: `<i>`жирный курсив`</i>`

Правильно: HTML-код: `<i>`жирный курсив`</i>`

В браузере: ***жирный курсив***

Структура документа HTML

```
<html>
<head>
...
</head>
<body>
...
</body>
</html>
```

 } заголовок документа

 } тело документа

HTML-документ заключен в тег `<html>` и состоит из заголовка и тела. Заголовок документа лежит внутри тега `<head>` и содержит название документа и некоторые другие параметры. Тело документа заключено в тег `<body>` и содержит текст и теги, которые должен обработать и вывести браузер. Текст из тега `<title>` обычно отображается в заголовке окна браузера, а также в результатах поиска поисковых систем.

Пример: простейший HTML-документ

```
<html>
<head>
<title>Заголовок</title>
</head>
<body>
Мой первый <i>HTML-документ!</i><br>
(это пример)
</body>
</html>
```

Представление цвета в HTML

Цвет в HTML может быть задан ключевыми словами – названиями цветов на английском языке:

Таблица 2.1. Стандартные цвета HTML

Название в HTML	Название на русском	Код в RGB
aqua	морская волна	#00ffff
black	черный	#000000
blue	синий	#0000ff
fuchsia	фуксия	#ff00ff
grey	серый	#808080
green	зеленый	#008000
lime	ярко-зеленый	#00ff00
maroon	темно-бордовый	#800000
navy	темно-синий	#000080
olive	оливковый	#808000
purple	пурпурный	#800080
red	красный	#ff0000
silver	серебряный	#c0c0c0
teal	бирюзовый	#008080
white	белый	#ffffff
yellow	желтый	#ffff00

Но компьютер может отобразить гораздо больше – около 16 миллионов – цветов. Альтернативным способом задания цвета является указание кода цвета в системе RGB (от англ. *red*, *green*, *blue* – красный, зеленый, синий). Суть системы заключается в том, что любой цвет может быть представлен как смешение основных цветов – красного, зеленого и синего. Цвет записывается в виде 6-символьного кода.

Код представляет собой шестнадцатеричное число от 000000 до FFFFFFFF. Первые две цифры соответствуют красной компоненте, следующие две – зеленой, последние две – синей. Значение 00 означает полное отсутствие составляющей, значение FF (255) – максимум составляющей. В качестве шестнадцатеричных цифр используются десятичные цифры от 0 до 9 и латинские буквы от A до F для обозначения цифр от 10 до 15. Таким образом, получается $256^3 \approx 16.7$ млн. цветов – этого достаточно, чтобы воспроизвести любой цвет, который различает человеческий глаз.

Например:

- FF0000 – ярко-красный (red)
- 00FF00 – ярко-зеленый (green)
- 0000FF – ярко-синий (blue)
- FFFF00 – желтый (yellow) – смесь красного и зеленого
- 000000 – черный (black)
- FFFFFF – белый (white)

Значение цвета указывается в теге после символа решетки (#).

Например для текста:

```
<font color="#808080">серый текст</font>
```

Для фона всей страницы в теге body атрибут bgcolor:

```
<body bgcolor="#FFFF00">желтый фон</body>
```

Значение кода сложно подобрать самому, поэтому используются специальные инструменты.

Например, можно набрать в Яндексе запрос «подбор цвета»:



Рисунок 2.1 Инструмент подбора цвета в Яндексе

Подобные инструменты есть во всех графических редакторах:

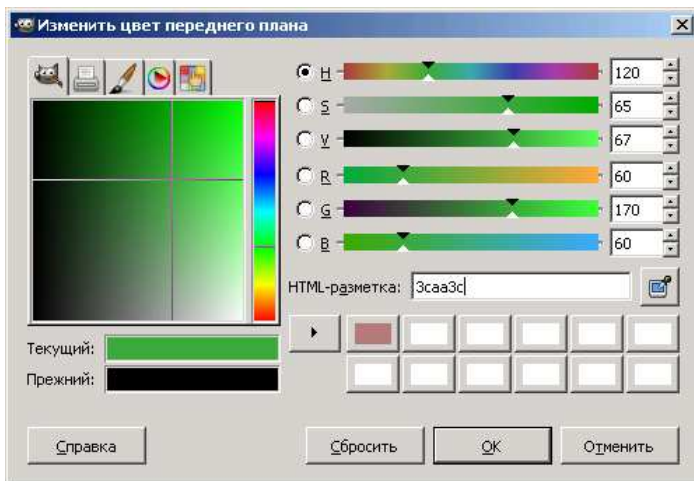




Рисунок 2.2. Диалог выбора цвета в редакторе GIMP.

 Продемонстрируйте выбор цвета в редакторе, который предполагается использовать в курсе.

Не используйте кириллические названия файлов и папок – давайте им английские названия! Первая страница всегда носит название `index.html`.

 Ресурсы в Интернете

- Создание HTML-документа. <http://stepbystep.htmlbook.ru/?id=2>
- Особенности HTML. <http://stepbystep.htmlbook.ru/?id=3>
- Структура HTML-кода. <http://stepbystep.htmlbook.ru/?id=4>
- Теги HTML. <http://stepbystep.htmlbook.ru/?id=5>



Задания:

а) Создайте HTML-страницу со следующим содержанием:

Мой первый сайт!

(это пример)

Фамилия Имя Отчество

Название страницы – «Моя первая страница». Фамилия, имя и отчество должны выводиться разными цветами. Название файла – index.html. Задайте странице цветной фон: подберите такой цвет, чтобы он не затруднял чтение текста.

б*) Создайте HTML-страницу с фразой: «Каждый Охотник Желает Знать Где Сидит Фазан». Каждое слово должно быть соответствующего цвета: красный, оранжевый, желтый, зеленый, голубой, синий, фиолетовый. Дайте странице заголовок «Радуга».

У к а з а н и е: для получения кодов цветов используйте подбор цвета в Яндексе или подобный инструмент.

Лекция 2.2. Основные теги, работа с текстом, списки

В стандарте HTML 4.01 перечислен 91 тег. Каждый тег предназначен для решения определенной задачи: работы с текстом, ссылками, графикой, таблицами и т.д. С некоторыми из тегов мы уже познакомились в предыдущей лекции.

Теги структуры документа

Эти теги предназначены для определения структуры HTML документа и не влияют на его отображение в браузере. Тем не менее, правильно сформированный документ обязательно должен их содержать.

`<html>...</html>` – включает в себя все содержимое веб-страницы, в том числе теги `<head>` и `<body>`

`<head>...</head>` – содержит теги со служебной информацией о странице, например название в теге `<title>`.

`<title>...</title>` – задает название документа. Это название обычно отображается в заголовке окна браузера.

`<body>...</body>` – хранит содержимое документа.

Атрибуты:

`bgcolor="цвет"` – назначает цвет фона документа

`text="цвет"` – указывает цвет обычного текста в документе

! Теги `<html>`, `<head>`, `<title>` и `<body>` задаются в документе только 1 раз!

Теги для работы с текстом

HTML позволяет управлять отображением текста на странице.

`...` – выделение текста жирным

`<i>...</i>` – выделение текста курсивом

`<u>...</u>` – подчеркивание текста

`_{...}` – форматировать текст как подстрочный индекс

Пример:

HTML-код: `101₂` = 5

В браузере: `1012` = 5

`^{...}` – форматировать текст как надстрочный индекс

Пример:

HTML-код: `2⁸` = 256

В браузере: `28` = 256

`<center>...</center>` – выравнивание текста по центру

... – устанавливает размер, цвет и гарнитуру текста

Атрибуты:

color="цвет" – задает цвет текста

face="шрифт" – определяет гарнитуру текста; значением атрибута может быть список шрифтов, перечисленных через запятую – в этом случае выбирается первый доступный шрифт

size="1-7" – устанавливает размер шрифта (от 1 до 7)

Пример:

HTML-код:

```
<FONT face="Tahoma" size="2" color="gray">текст</FONT>
```

В браузере: ТЕКСТ

<p>...</p> – задает начало и конец параграфа

Атрибут:

align="..." – определяет режим выравнивания текста

left – по левому краю (по умолчанию)

center – по центру

right – по правому краю

justify – по ширине

<hN>...</hN> – вложенный текст, является заголовком документа уровня N, N принимает значения от 1 до 6. Наибольшим заголовком является <h1>, наименьшим <h6>.

**
** – перенос строки (см. Лк №1)

Тег HR

<hr> – выводит горизонтальную разделительную линию

Атрибуты:

align="..." – определяет режим выравнивания линии

left – по левому краю

center – по центру (по умолчанию)

right – по правому краю

noshade – использовать сплошную линию вместо объемной

size="N" – толщина линии в пикселах

`width="N"` – ширина линии в пикселах или процентах по отношению к ширине экрана.

Размеры объектов в HTML часто указываются в пикселях. Пиксель – наименьший элемент изображения на экране (точка). Количество пикселей на экране по горизонтали и вертикали называют разрешением (например, 1024 по горизонтали на 768 по вертикали).

Работа со списками

В HTML есть возможность создавать нумерованные и маркированные списки.

`...` – создает нумерованный список элементов

Атрибуты:

`start="N"` – начать нумерацию с числа N

`type="..."` -определяет формат нумерации

1 – арабские цифры (по умолчанию)

A – прописные буквы (A, B, C)

a – строчные буквы (a, b, c)

I – прописные римские цифры (I, II, III)

i – строчные римские цифры (i, ii, iii)

`...` – создает маркированный список элементов

Атрибут:

`type="..."` – определяет формат маркера

disk – диск (по умолчанию)

circle – окружность

square – квадрат

`...` – задает элемент списка в нумерованном или маркированном списке

Атрибуты:

`type="..."` – формат номера или маркера (см. описание `` и ``)

`value="N"` – задает номер элемента списка

Пример:

HTML-код:

```
<ol>
<li>арабские цифры (по умолчанию)</li>
<li type="A">прописные буквы</li>
<li type="a">строчные буквы</li>
<li type="I">прописные римские цифры</li>
<li type="i">строчные римские цифры</li>
</ol>
<ul>
<li>диск (по умолчанию)</li>
<li type="circle">окружность</li>
<li type="square">квадрат</li>
</ul>
```

В браузере:

- I. арабские цифры (по умолчанию)
 - В. прописные буквы
 - с. строчные буквы
 - IV. прописные римские цифры
 - v. строчные римские цифры
- диск (по умолчанию)
 - окружность
 - квадрат



Ресурсы в Интернете

- Работа с текстом. <http://stepbystep.htmlbook.ru/?id=6>
- Списки. <http://stepbystep.htmlbook.ru/?id=10>



Задания:

а) Создайте страницу «Мое хобби». Страница должна содержать заголовок «Мое хобби», выровненный по центру, краткое описание вашего хобби и нумерованный список ваших интересов (спорт, науки, игры и т.п.). Название файла – hobby.html.

- б) Измените тип нумерации на нумерацию буквами и римскими цифрами.
- в) Измените тип списка на маркированный, используйте разные типы маркеров.
- г*) Создайте текстовую надпись большого размера. Примените к ней по очереди шрифты Wingdings, Wingdings 2, Wingdings 3 и Webdings.

Лекция 2.3. Создание ссылок

Для создания ссылок используется тег `<a>...`.

Обязательный атрибут `href` указывает абсолютный или относительный адрес, на который ведет ссылка. Ссылка может указывать на HTML-документ, изображение, файл для сохранения на диск и пр. Текст ссылки записывается между открывающим и закрывающим тегом.

Абсолютный адрес содержит в себе имя хоста и полный путь к ресурсу, например: `http://www.example.com/docs/about.html`. С помощью абсолютного адреса можно сослаться на любой открытый ресурс в Интернете. Если нужно поставить ссылку на главную страницу сайта, указывают его адрес и слеш.

Пример для абсолютного адреса:

HTML-код: `Яндекс`

В браузере: [Яндекс](#)

В браузере ссылка обычно представляется как подчеркнутый текст. При клике по ссылке браузер загружает страницу, указанную в атрибуте `href`.

Также для документов, расположенных на том же сайте, можно использовать относительный адрес.



Рисунок 2.3. Пример файловой структуры

Например, чтобы поставить ссылку из файла file1.html на файл file2.html (см. рис. 2.3.), необходим следующий HTML-код:

```
<A href="folder1/file2.html">файл file2.html</A>
```

А чтобы ссылка в файле file2.html указывала на file1.html:

```
<A href="../file1.html">файл file1.html</A>
```

Две точки (..) означают переход к родительскому каталогу.

Замечание: для файлов в пределах одного сайта рекомендуется использовать только относительные пути. В этом случае ссылки сохранят работоспособность при изменении адреса сайта, переносе в другую папку и т.п.

Для открытия ссылки в новом окне используется атрибут `target` со значением `_blank`.

Пример: `Яндекс`

Цвет ссылок в документе можно указать атрибутами тега `<body>`:

`alink="цвет"` – устанавливает цвет активных ссылок

`link="цвет"` – задает цвет непосещенных ссылок

`vlink="цвет"` – определяет цвет посещенных ссылок



Ресурсы в Интернете

- Ссылки. <http://stepbystep.htmlbook.ru/?id=7>
- Гипертекстовые ссылки.
<http://www.intuit.ru/department/internet/htmlbasics/3/1.html>
- «Якоря»*. <http://stepbystep.htmlbook.ru/?id=8>
- Ссылки на e-mail*. <http://www.computerra.ru/gid/rtfm/mail/35658/>



Задания:

а) Модифицируйте файл `index.html`: добавьте ссылку на страницу «Мое хобби» и ссылку на сайт отдела (должна открываться в новом окне). На странице «Мое хобби» добавьте гиперссылку, указывающую на страницу `index.html`.

б*) Создайте страницу links.html. Разместите на ней ссылки на ваши любимые сайты. Ссылки должны быть расположены в нумерованном или маркированном списке и открываться в новом окне.

Лекция 2.4. Изображения

Вставка изображений на странице

Осуществляется непарным тегом ``. Обязательный атрибут `src` указывает абсолютный или относительный URL изображения (см. Лк 2.3.). Стандартными форматами изображений являются GIF, PNG и JPEG.

GIF – формат, реализующий сжатие без потери качества с ограниченной цветностью (от 2 до 256 цветов) и поддержкой анимации – используется для хранения графики, когда достаточно 256 (и меньше) цветов. Обычно это небольшие изображения. Также GIF поддерживает прозрачность.

JPEG реализует сжатие изображений с потерями качества, при этом ограничения на цвет отсутствуют (поддерживается 16 миллионов цветов). Размер JPEG-файла зависит от параметра «качество», который указывается при его сохранении: от 0 до 100. Чем выше качество, тем больше размер файла. Оптимальная степень качества зависит от изображения, в большинстве случаев она равна 70-80. Не стоит выставлять этот параметр меньше 50 – на изображении появятся заметные дефекты или больше 95 – размер файла сильно возрастет без видимого улучшения качества.

Формат **PNG** существует в двух вариантах: PNG-8 и PNG-24. PNG-8, как и GIF, поддерживает 256 цветов, обеспечивает по сравнению с ним лучшее сжатие, но не поддерживает анимацию. Формат PNG-24, как и JPEG, не имеет ограничений на количество цветов, но проигрывает ему в размере файла. Осуществляет сжатие изображений без потери качества, поэтому его стоит применять для изображений, содержащих мелкие детали.

Ниже приведена таблица оптимального выбора формата файла в соотношении качество/размер файла с учетом особенностей изображения.

Таблица 2.2. Выбор формата графического файла

Особенности изображения	Предпочтительный формат
анимированное изображение	только GIF
маленькое изображение с небольшим количеством цветов	GIF или PNG-8
изображение с полупрозрачностью	только PNG
изображение с большим количеством цветов, например фотография	JPEG
изображение с большим количеством цветов с мелкими деталями, например скриншот (снимок экрана)	PNG-24

Избегайте использования других форматов изображений (например, BMP или TIFF), т.к. они могут не поддерживаться отдельными типами браузеров.

Другие атрибуты:

`align="..."` – определяет режим выравнивания изображения относительно текста в строке:

`top` – по верхнему краю

`middle` – по центру строки

`bottom` – по нижнему краю (по умолчанию)

`left` – по левому краю окна

`right` – по правому краю окна

`alt="..."` – определяет текст, описывающий изображение для браузеров без поддержки графики (или с отключенной графикой), поисковых машин и т.п.

`border="N"` – устанавливает толщину рамки вокруг изображений, равной N пикселей, 0 – для отключения рамки

`height="N"` – высота изображения в пикселях или процентах

`width="N"` – ширина изображения в пикселях или процентах

Браузер определяет размер изображения автоматически. Для ускорения загрузки рекомендуется указывать размер изображения атрибутами `height` и `width`, чтобы браузер не вычислял этот размер автоматически после загрузки изображения. Также этими атрибутами можно растянуть/сжать изображение

по горизонтали/вертикали, но такое масштабирование приведет к потере качества.

Изображение может быть сделано ссылкой, путем помещения внутрь тега `<a>`. В этом случае вокруг изображения автоматически появляется рамка. Толщина рамки задается атрибутом `border`. Обычно рамку убирают, указывая `border="0"` в теге ``.

Примеры:

1. HTML-страница находится в папке `site`, а изображение `picture.jpg` находится в папке `site/images/`.

```

```

2. Изображение находится на другом сайте в Интернет

```

```

Фоновое изображение страницы

Можно задавать адрес фонового изображения для страницы в атрибуте `background` тега `<body>`. Фоновое изображение отображается в натуральную величину. Если размер изображения меньше размера окна браузера, то рисунок повторяется по горизонтали вправо и по вертикали вниз. Например, зададим фоновым изображением страницы рисунок `bg1.jpg`.

Тест!

Рисунок. 2.4. Файл `bg1.jpg`

```
<html>
<head>
<title>Тест фона</title>
</head>
<body background="bg1.jpg">
</body>
</html>
```

В браузере:

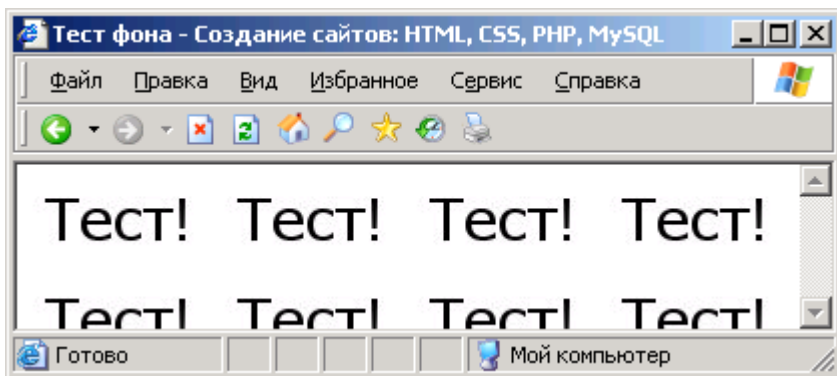


Рисунок 2.5. Размножение фонового рисунка в браузере

Чтобы сделать неповторяющийся фон, необходимо выбрать картинку заведомо большую, чем размер страницы по ширине и высоте.

Можно использовать повторяющийся фон:



Рисунок 2.6. «Шахматный» фон

```
<html>
<head>
<title>Тест фона №2</title>
</head>
<body background="checkmate.gif">
Страница с шахматным фоном.
</body>
</html>
```

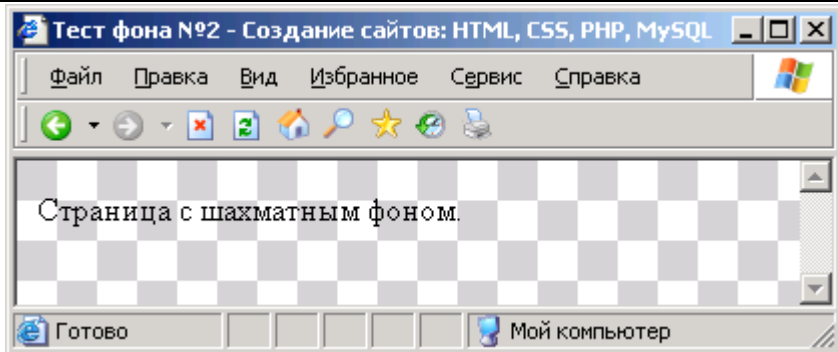


Рисунок 2.7. Использование повторяющегося «шахматного» фона

Если взять картинку шириной 1 пиксель и высотой, например, 2000 пикселей на экране она будет размножаться только по горизонтали.

```
<html>
<head>
<title>Тест фона</title>
</head>
<body background="1px.gif">
<h1><font color="white" face="Arial">Фон</font></h1>
Используем картинку с градиентом!
</body>
</html>
```

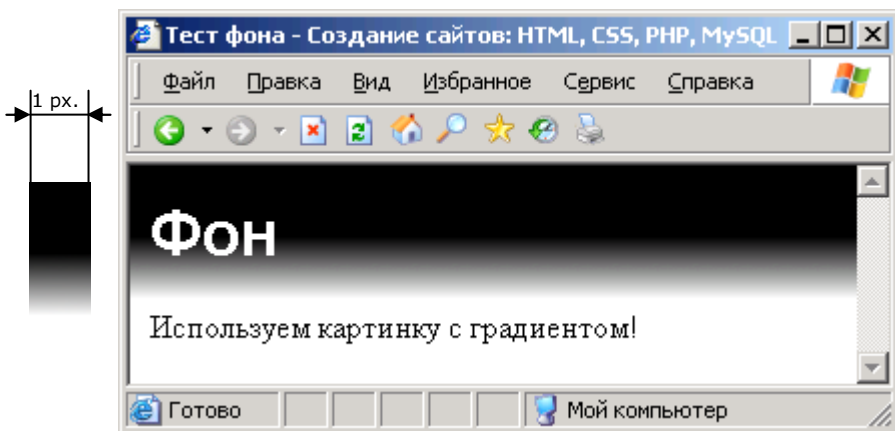


Рисунок 2.8. Использование изображения, повторяющегося по горизонтали

При использовании изображений в качестве фона важно обеспечить читабельность текста на странице. Поэтому не стоит применять фотографии в качестве фоновых картинок.



Ресурсы в Интернете

- Изображения. <http://stepbystep.htmlbook.ru/?id=9>
- Изображения в HTML.
<http://www.intuit.ru/department/internet/htmlbasics/7/>
- Форматы графических изображений для WEB.
http://www.grafika-online.com/statia/art04_web/art01.html
- Различие между PNG и JPEG. <http://bit.ly/62Ek2O>



Задания:

а) Добавьте на первую страницу (index.html) свою фотографию вместо строк «Мой первый сайт, это пример».

б) При помощи атрибутов width и height уменьшите и увеличьте размер изображения в 2 раза. Обратите внимание на потерю качества изображения при увеличении.

в) Сделайте изображение на первой странице гиперссылкой: при нажатии на фотографию должен открываться полноразмерный вариант в новом окне.

г) Добавьте графический фон на страницы сайта.

д*) Добавьте на страницу информер (небольшая картинка, показывающая погоду, курс валют и т.п. актуальную информацию). URL информера можно найти в поисковой системе или на специализированном сайте. Например: <http://www.informer.ru/> , <http://gismeteo.ru/>

Лекция 2.5 Создание таблиц

Таблица в HTML – это совокупность данных, расположенных и связанных между собой при помощи ячеек, размещаемых в строках и колонках. Таблица заполняется данными построчно. Для вставки таблиц определено 3 основных тега. Содержимое ячеек помещается в теги `<td>...</td>`, которые, в свою очередь, помещаются в теги строк `<tr>...</tr>`, а они уже – в тег `<table>...</table>`.

Пример:

```
<table>
<tr> <td>1</td> <td>2</td> <td>3</td> </tr>
<tr> <td>4</td> <td>5</td> <td>6</td> </tr>
</table>
```

В браузере:

1	2	3
4	5	6

Количество тегов `<tr>...</tr>` определяет количество строк. В каждом теге строки должно быть одно и то же число тегов `<td>...</td>`, равное числу столбцов, иначе таблица отобразится неправильно. Можно создавать вложенные таблицы: вкладывать таблицу в ячейку другой таблицы.

`<table>...</table>` – определяет начало и конец кода таблицы, содержит в себе теги строк и ячеек.

Атрибуты:

`align="..."` – определяет режим выравнивания таблицы относительно текста в строке
`left` – по левому краю

`right` – по правому краю

`background="URL"` – задает фоновый рисунок в таблице

`bgcolor="цвет"` – цвет фона таблицы

`border="N"` – устанавливает толщину границ таблицы, равную N пикселей
(0 для отключения)

`bordercolor="цвет"` – цвет рамки

`cellpadding="N"` – размер поля вокруг содержимого каждой ячейки

Пример:

`cellpadding="0"`

1	2
3	4

`cellpadding="15"`

1	2
3	4

`cellspacing="N"` – размер свободного пространства между ячейками

Пример:

`cellspacing="0"`

1	2
3	4

`cellspacing="15"`

1	2
3	4

`width="N"` – ширина таблицы в пикселях или процентах от ширины окна

`<tr>...</tr>` – определяет строку ячеек таблицы

Атрибуты:

`align="..."` – определяет режим выравнивания содержимого ячеек строки

`left` – по левому краю

`center` – по центру

`right` – по правому краю

`justify` – по ширине

`background="URL"` – URL изображения, которое заполнит фон ячеек строки

`bgcolor="цвет"` – цвет фона ячеек строки

`valign="..."` – определяет режим выравнивания содержимого ячеек строки по вертикали

`top` – по верхнему краю

`middle` – по середине (по умолчанию)

`bottom` – по нижнему краю

`<td>...</td>` – определяет ячейку данных таблицы

Атрибуты:

`align="..."` – определяет режим выравнивания содержимого ячейки

`left` – по левому краю

`center` – по центру

`right` – по правому краю

`background="URL"` – URL изображения, которое заполнит фон ячейки

`bgcolor="цвет"` – цвет фона ячейки

`valign="..."` – определяет режим выравнивания содержимого ячейки по вертикали

`top` – по верхнему краю

`middle` – по середине (по умолчанию)

`bottom` – по нижнему краю

`height="N"` – высота ячейки в пикселях

`width="N"` – ширина ячейки в пикселях или процентах от ширины таблицы

Объединение ячеек

`colspan="N"` – растягивает ячейку на N столбцов влево

Пример:

HTML-код:

```
<TABLE cellpadding="15" border="1">
<TR><TD colspan="2">1</TD></TR>
<TR><TD>2</TD><TD>3</TD></TR>
</TABLE>
```

В браузере:

1	
2	3

`rowspan="N"` – растягивает ячейку на N строк вниз

Пример:

HTML-код:

```
<TABLE cellpadding="15" border="1">
<TR><TD rowspan="2">1</TD><TD>2</TD></TR>
<TR><TD>3</TD></TR>
</TABLE>
```

В браузере:

1	2
	3

Ширина таблицы

Если ширина таблицы изначально не задана, то она вычисляется исходя из содержимого ячеек.

```
<table border="1">
<tr><td>Ширина таблицы не задана!</td></tr>
</table>
```

Ширина таблицы не задана!

Максимальная ширина таблицы в таком случае равна ширине окна.

Если же ширина задана атрибутом `width`, то браузер расставляет переносы слов в тексте ячеек таким образом, чтобы соблюсти заданный размер.

```
<table width="100" border="1">
<tr><td>Если задать атрибут width, текст начинает
переноситься по словам</td></tr>
</table>
```

Если задать
атрибут
`width`, текст
начинает пе-
реноситься
по словам



Ресурсы в Интернете

- Таблицы. <http://stepbystep.htmlbook.ru/?id=11>
- Таблицы. <http://www.intuit.ru/department/internet/htmlbasics/4/>



Задания:

а) Создайте следующую таблицу:

1	2	
	3	4

б) Добавьте страницу «Мой компьютер» (computer.html), содержащую заголовок страницы и таблицу данных о вашем компьютере (аппаратное обеспечение: процессор, объем оперативной памяти и т.п., операционная система) и ссылку на главную страницу. Если дома компьютера нет или не известны характеристики, опишите компьютер в классе.

Добавьте ссылку на computer.html на главной странице.

У к а з а н и е: Параметры процессора и памяти можно посмотреть в свойствах пиктограммы «Мой компьютер», объем жестких дисков – в «Моем компьютере».

Пример:

Процессор	Intel Core2 Duo 2.66 ГГц
Оперативная память	2 Гб
Жесткий диск	250 Гб
Подключение к интернет	10 Мбит/с
Операционная система	Windows XP Professional

в) * Создайте таблицу основных цветов палитры RGB и их комбинаций. Желтый, пурпурный и бирюзовый цвета получаются при смешивании красного, зеленого и синего друг с другом. Фон каждой ячейки должен соответствовать указанному цвету.

-	Красный FF0000	Зеленый 00FF00	Синий 0000FF
Красный FF0000	Красный FF0000	Желтый FFFF00	Пурпурный FF00FF
Зеленый FF0000	Желтый FFFF00	Зеленый 00FF00	Бирюзовый FF00FF
Синий 0000FF	Пурпурный FFFF00	Бирюзовый 00FF00	Синий FF00FF

Лекция 2.6. Кодировки текста и специальные символы



Лекция не является обязательной.

Однobaйтные и многобайтные кодировки

Все данные в компьютере: текст, графика, звук, видео – хранятся и обрабатываются в цифровой (двоичной) форме. Минимальной единицей измерения информации является бит – разряд двоичного числа, который может принимать 2 значения: 0 или 1. 8 бит составляют 1 байт – минимальную единицу адресации к хранимой информации.

Текст представляет из себя набор символов: букв, цифр, знаков препинания и других. Для работы с текстом компьютер должен представить его в «привычном» для себя виде: нулями и единицами. Для этого задаются таблицы соответствия символа и его числового кода – таблицы кодировки.

В 1960-гг. в США была создана кодировка ASCII. Числовой код символа записывался 7-ю битами, таким образом, можно было составлять тексты, используя $2^7 = 128$ различных символов. Кодировка содержит управляющие символы (например, перенос строки, табуляция), латинские символы, цифры, знаки пунктуации и др. Так, пробелу соответствовал код 32, цифре 0 – код 48, заглавной латинской букве А – код 65, а строчной – 97.

ASCII в своей 7-битной версии не позволяет использовать символы национальных алфавитов, например кириллицу: для нее не осталось места в таблице. Так как компьютеры работают с байтом из 8 бит, в ASCII использовались только первые 7 бит, а последний бит всегда был равен 0. Задействовав этот последний бит, можно получить еще дополнительно 128 мест. Таким способом ASCII была дополнена алфавитами для различных языков. Для каждого языка создавались кодировки, первые 128 символов которых повторяли ASCII, а во второй половине таблицы кодировки располагались символы национальных алфавитов. К сожалению, конкуренция между разработчиками операционных систем привела к появлению нескольких несовместимых друг с другом кодировок для кириллицы: КОИ-8 в UNIX, CP866 в MS-DOS, Windows-1251 в Windows, MacCyrillic на компьютерах «Макинтош» фирмы Apple. Чтобы устранить «зоопарк» кодировок, Международной организацией по стандартизации (ISO) была разработана единая кодировка ISO 8859-5, но она так и не прижилась в компьютерной индустрии. Однобайтные кодировки обладают следующими недостатками.

- Документы, созданные в одной кодировке, отображаются как беспорядочный набор символов в другой
- Использование однобайтной кодировки ограничивает набор 256-ю символами (на самом деле, еще меньше – первые 32 символа являются служебными). Это делает невозможным работу с иероглифическими языками (китайским, японским), в которых используются тысячи различных знаков, а также использование множества языков в одном документе.

Для решения этих проблем был создан стандарт кодирования Unicode (Юникод), который содержит символы практически всех существующих письменных языков и изобретенных человечеством знаков (музыкальных, математических и т.п.). Юникод устраняет проблему выбора правильной кодировки, но текст, сохраненный в этой системе, занимает больший объем байт. Существует несколько представлений Unicode: UTF-16, где на 1 символ приходится 2 байта, и UTF-8 с переменным числом байтов на символ – от 1 до 4. Если на странице используется только латиница и кириллица, то каждый символ для хранения будет занимать 2 байта, т.е. текст такой HTML-страницы в UTF-8 требует в 2 раза больше места на диске, чем текст в кодировке Windows-1251. При нынешних темпах развития систем хранения данных это уже

можно не считать серьезным недостатком. На начало 2010 года UTF-8 используется более чем на 50% сайтов, а кодировки ASCII, Windows-1251 и прочие из года в год используются веб-мастерами все реже.

Кодировка в HTML

Кодировка документа HTML задается в текстовом редакторе. Например, Блокнот в ОС Windows по умолчанию сохраняет текстовые файлы в кодировке Windows-1251.



Продемонстрируйте выбор кодировки в используемом на занятиях текстовом редакторе.

Для того чтобы браузер правильно отобразил HTML-страницу, необходимо задать правильную кодировку в специальном теге `<meta>`¹.

```
<meta http-equiv="Content-Type" content="text/html ; charset=windows-1251">
```

или

```
<meta http-equiv="Content-Type" content="text/html ; charset=utf-8">
```

Если кодировка не будет указана, браузер попытается «угадать» ее, но не всегда это заканчивается успехом. Пользователь может выбрать кодировку самостоятельно в меню браузера (в Internet Explorer и Mozilla Firefox: Вид → Кодировка). При разработке сайта проблем с кодировкой следует избегать, т.к. большинство пользователей сразу же покинет страницу, увидев нечитаемый набор букв на экране.

Специальные символы в HTML

В HTML предусмотрен механизм вставки в документ любых символов Юникод – подстановки или сущности (англ. entities). Подстановки позволяют употреблять символы, отсутствующие на клавиатуре или даже в используе-

¹ Другой способ указания кодировки будет рассмотрен в теме 9. Назначение тега `<meta>` описано в теме 10.

мой кодировке (т.е. даже используя кодировку Windows-1251 можно вставить букву греческого алфавита). Подстановки начинаются с символа амперсанда и записываются в виде `&#DDDD;` где DDDD – код символа в Юникоде в десятичной системе счисления. Также можно записывать код в шестнадцатеричной системе счисления в форме `&#xHHHH;` Для некоторых символов заданы специальные названия – мнемоники. Например, знак копирайта © может быть задан кодом `©` или `©` или мнемоникой `©`.

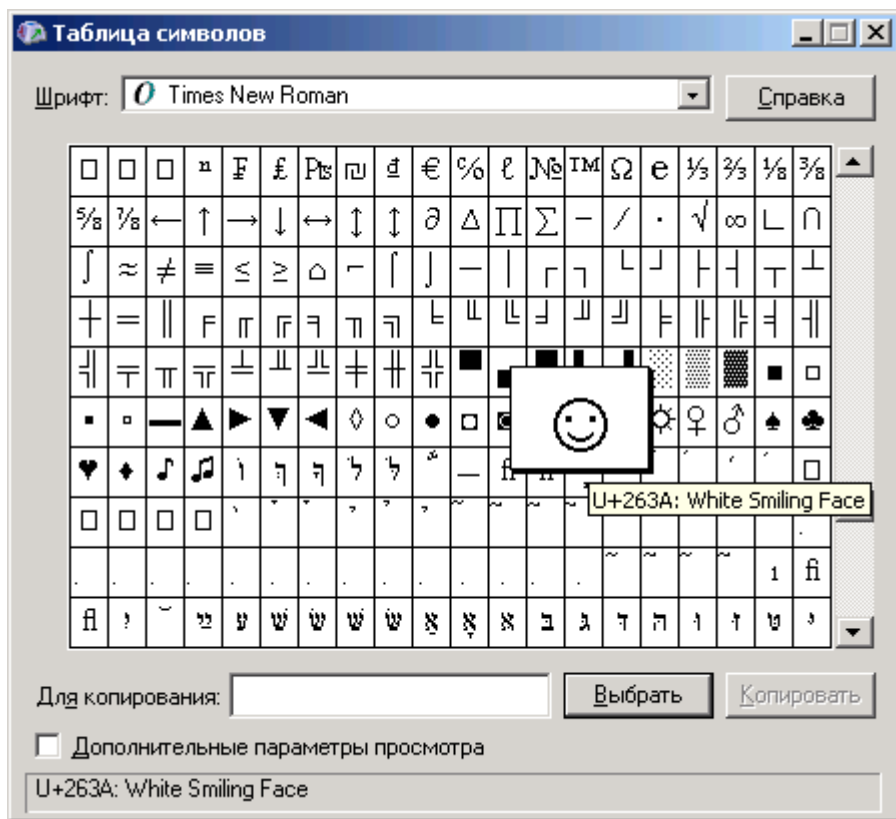


Рисунок 2.10. Интерфейс программы «Таблица символов»

Для того чтобы найти код нужного символа в ОС Windows, можно воспользоваться системной утилитой «Таблица символов» (см. рис. 2.10). Открыть программу можно нажав кнопку Пуск → Выполнить → `charmap`.

В программе можно посмотреть, какие символы поддерживают установленные на компьютере шрифты, и узнать шестнадцатеричные коды этих символов.

Часто используются подстановки:

`&lquo;` и `&rquo;` – для кавычек «елочек»;

`—` – для тире;

неразрывный пробел ` `; (см. список Интернет-ресурсов в конце лекции)

`<` и `>` – для символов меньше (<) и больше (>), которые используются для задания тегов HTML.

В приложении 3 приведена таблица других часто используемых подстановок.



Ресурсы в Интернете

- О кодировках символов. <http://www.arinnav.ru/js/encoding.htm>
- Мнемоники в HTML. http://ru.wikipedia.org/wiki/Мнемоники_в_HTML
- Пробел. <http://ru.wikipedia.org/wiki/Пробел>
- Кавычки. <http://www.artlebedev.ru/kovodstvo/sections/104/>
- Тире, минус и дефис, или Черты русской типографики. <http://www.artlebedev.ru/kovodstvo/sections/97/>



Задания:

а) Найдите код и вставьте в HTML-документ символы ☀️ ❤️ ≈ € 🇷🇺

б*) Составьте формулу объема конуса: $V_{\text{кон}} = \frac{1}{3} \cdot \pi \cdot r^2 \cdot h$

Лекция 3.1. Основы CSS

CSS (Cascading Style Sheets – каскадные таблицы стилей, произносится «си-эс-эс») – технология управления внешним видом элементов (тегов) веб-страницы. CSS предоставляет гораздо больше возможностей по оформлению страницы, чем HTML. Например, с помощью стилей CSS можно убрать у ссылок подчеркивание, сделать у таблицы пунктирные границы или даже поменять курсор «мыши». Сейчас CSS используется практически на всех сайтах Всемирной паутины.

Синтаксис CSS

Рассмотрим синтаксис CSS. В стилях задается набор правил отображения в парах «свойство – значение», и то, к каким элементам их применять (селектор):

```
Селектор
{
    свойство1: значение1;
    свойство2: значение2;
    свойство3: значение3 значение4;
}
```

Правила записываются внутри фигурных скобок и отделяются друг от друга точкой с запятой. Между свойствами и их значениями ставится двоеточие.

CSS, как и HTML, игнорирует пробелы. Можно добавлять комментарии, заключая их между /* и */.

Селекторы

Селектор определяет, к каким элементам (тегам) страницы будут применяться правила, заданные парами «свойство – значение».

В качестве селектора можно использовать:

- Название тега – тогда стиль применится ко всем таким тегам.

Пример:

```
A {font-size: 12pt; text-decoration: none}
TABLE {border: black solid 1px}
```

Первая строчка этого CSS-кода задает всем ссылкам 12-й размер шрифта и убирает подчеркивание. На второй строчке указывается, что у всех таблиц граница будет черного цвета, сплошной (solid) и шириной 1 пиксель.

- Несколько тегов через запятую – тогда стиль применится для всех перечисленных тегов.

Пример:

```
h1, h2, h3, h4, h5, h6 {color: red} /* делаем все
заголовки красными */
```

- Несколько тегов через пробел:

```
TABLE A {font-size: 120%}
```

Правило относится ко всем тегам A, вложенным в тег TABLE. Размер шрифта увеличится на 20% от базового.

- ID элемента. В стилях уникальный идентификатор указывается после знака # – правила применяются к тегу с атрибутом

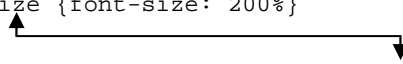
id="идентификатор". Пример:

CSS

```
#supersize {font-size: 200%}
```

HTML

```
<a href="http://htmlbook.ru" id="supersize">Справочник
HTML и CSS</a>
```



Нельзя вносить в документ несколько элементов с одинаковым id!

- Символ * – правила применяются ко всем элементам документа.
- Классы

Классы

Часто нужно, чтобы стиль применялся не ко всем тегам на странице, а только к некоторым элементам (например, не ко всем ссылкам на странице, а только к тем, которые расположены в меню сайта). Для этого используются классы:

```
ТЕГ.имя_класса { ... }
```

Правила, указанные после такого селектора, будут действовать только на теги с атрибутом `class="имя_класса"` :

```
<ТЕГ class="имя_класса"> ... </ТЕГ>
```

Можно не указывать имя тега, тогда правила будут применяться ко всем тегам с подходящим значением атрибута `class`.

Рассмотрим пример:

Для всех тегов с атрибутом `class="class1"` добавим подчеркивание текста и уменьшим размер шрифта, а для тега `` уберем подчеркивание.

```
.class1 {text-decoration: underline; font-size: 80%}
```

```
A.class1 {text-decoration: none;}
```

В HTML-коде укажем для тегов имя класса:

```
<h1 class="class1">Мои любимые сайты</h1>  
<a href="http://yandex.ru" class="class1">  
Яндекс</a><br>  
<a href="http://google.com" class="class1">  
Google</a><br>  
<a href="http://redut.ru" class="class1">Redut.ru</a>
```

В браузере будет отображаться:

Мои любимые сайты

[Яндекс](#)

[Google](#)

[Redut.ru](#)

Можно указывать для одного элемента несколько классов через пробел.

Стили CSS могут включаться в HTML-документ 3 разными способами:

Внешние стили.

Хранятся в отдельном файле .css. Подключаются тегом `<link rel="stylesheet" type="text/css" href="адрес_стиля">`

Основное преимущество: один стиль может использоваться сразу в нескольких документах HTML.

Во внешних файлах нужно хранить стили, общие для всего сайта, они влияют сразу на множество тегов во множестве документов. Это становится очень удобным, если сайт содержит много страниц. Например, мы хотим поменять на всех страницах сайта цвет фона и размер шрифта. Если все страницы подключают один и тот же внешний стиль CSS, достаточно в нем задать новый цвет фона и размер шрифта. Иначе придется редактировать каждую страницу отдельно. Если на сайте несколько десятков или сотен страниц это становится очень трудоемкой задачей.

CSS-файл может находиться и на другом сайте – в этом случае необходимо указать его абсолютный URL-адрес.

Реализуем наш предыдущий пример. Создадим файл `style.css`:

```
.class1 {text-decoration: underline; font-size: 80%}  
A.class1 {text-decoration: none;}
```

Теперь создадим саму страницу `links.html`:

```
<html>  
<head>  
<link rel="stylesheet" type="text/css" href="style.css">  
</head>  
  
<body>  
  
<h1 class="class1">Мои любимые сайты</h1>  
<a href="http://yandex.ru" class="class1">  
Яндекс</a><br>  
<a href="http://google.com" class="class1">  
Google</a><br>  
<a href="http://redut.ru" class="class1"> Redut.ru </a>
```

```
</body>
</html>
```

При открытии этой страницы браузер клиента загрузит также файл `style.css` и применит правила CSS к документу.



Обратите внимание: с помощью CSS можно отключить у ссылок подчеркивание. Средствами HTML этого сделать невозможно. **CSS значительно расширяет возможности оформления страницы.**

Второй важный момент: **использование CSS позволяет разделить оформление и содержимое документа.** В нашем примере правила оформления содержатся в файле `style.css`, а содержание – в `links.html`. Такое разделение существенно упрощает редактирование сайта в дальнейшем. **Рекомендуется для оформления использовать только средства CSS, отказаться от использования таких тегов, как ``, `<s>`, `<u>`, `<center>`, атрибутов `align`, `border`, `color`, `height`, `width` и т.д.**

Стили уровня документа

Применяются ко всему документу, записываются внутри тега `<style>...</style >`, который вкладывается в тег `<head>...</head>` в документе HTML. Такой способ указания стилей используется, когда нужно применить одинаковые стили сразу к множеству HTML-элементов (тегов) в одном документе.

Добавим в наш пример тег `<style>`:

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="style.css">
<style>

</style>
</head>

<body>
```

```
<h1 class="class1">Мои любимые сайты</h1>
<a href="http://yandex.ru" class="class1">
Яндекс</a><br>
<a href="http://google.com" class="class1">
Google</a><br>
<a href="http://redut.ru" class="class1"> Redut.ru </a>

</body>
</html>
```

Внутренние стили

Используются, когда нужно указать стили конкретного единственного элемента. Внутренний стиль записывается в атрибуте `style` и применяется только к содержимому этого тега. Внутренний стиль имеет более высокий приоритет, чем внешние стили и стиль уровня документа. Предпочтительно не использовать такой способ задания стиля, т.к. он не отвечает принципу разделения содержания и оформления.

Порядок применения стилей

При работе с CSS необходимо помнить, что более специфичные правила имеют приоритет над менее специфичными, например:

- стиль, указанный в атрибуте `style`, перекрывает стиль, указанный в теге `<style>` или внешнем файле CSS:

```
<html>
<head>
<style>
A {color: red; text-decoration: none}
</style>
</head>
<body>
<a href=http://intuit.ru style="color:
green">INTUIT</a>
</body>
</html>
```

В браузере ссылка будет неподчеркнутой, зеленого цвета.

- селектор ID (#) имеет больший приоритет, чем селектор класса (.), а тот, в свою очередь, – больший, чем обычный селектор тега:

```
<html>
<head>
<style>
A {color: red; text-decoration: none; font-size: 120%}
.links {color: blue; text-decoration: underline}
#greenlink {color: green}
</style>
</head>
<body>
<a href="http://htmlbook.ru" class="links"
id="greenlink">htmlbook.ru</a>
</body>
</html>
```

В браузере ссылка будет зеленой и подчеркнутой, размер шрифта увеличен на 20%.

Другой важной особенностью CSS является то, что некоторые атрибуты наследуются от родительского элемента к дочернему. Например, если атрибут `font-size` задан для тега `<body>`, то он наследуется всеми элементами на странице. Когда свойство размера задается в процентах, оно будет вычислено исходя из значения для родительского элемента. Узнать, является ли атрибут наследуемым, можно в справочнике по атрибутам CSS (например, <http://htmlbook.ru>).



Ресурсы в Интернете

- Основы CSS. http://css.manual.ru/articles/css_basics
- Основы CSS. <http://www.intuit.ru/department/internet/operawebst/27/>
- Наследование и каскадирование.
<http://www.intuit.ru/department/internet/operawebst/28/>
- Устаревшие (не рекомендуемые к использованию) теги и атрибуты HTML. http://www.tutorialspoint.com/html/html_deprecated_tags.htm
- CSS по шагам. <http://stepbystep.htmlbook.ru/?pid=5>
- CSS-селекторы*. <http://www.alexilin.ru/css-selektory/>



Задания:

- а) Создайте внешний CSS файл. Подключите его ко всем страницам вашего сайта. Увеличьте размер шрифта, задайте для тега BODY фон свойством `background-color` и границу толщиной 5px.
- б) На главной странице измените цвет фона на отличный от цвета на других страницах.
- в) Создайте 2 различных класса стилей для ссылок на внутренние страницы (в навигационном меню) и внешних ссылок. Добавьте атрибут `class` в теги `<a>` на страницах.
- г*) Сохраните на диск копию какой-либо страницы из Интернета. Отредактируйте ее код: добавьте границу для всех элементов страницы.

Лекция 3.2. CSS-свойства: размеры, цвета, шрифты, текст

Размеры

Размеры в CSS можно задавать в различных единицах измерения:

- **em** – текущая высота шрифта
- **pt** – пункты (типографская единица измерения шрифта)
- **px** – пиксель
- **%** – процент

Гораздо реже используется указание размеров в миллиметрах (mm), сантиметрах (cm) и дюймах (in).

Единица измерения записывается сразу за значением без пробела:

```
TABLE {font-size: 12pt}
```

Цвета

В CSS цвет задается как и в HTML – 6 шестнадцатеричными цифрами: по 2 на каждый базовый цвет (красный, зеленый, синий). Также можно использовать стандартные названия цветов на английском (см. лекцию 2.1). Например:

```
A.content {color: black}
```

```
A.menu {color: #3300AA}
```

Допускается сокращать шестнадцатеричное представление до 3 цифр: запись #3300AA можно заменить на #30A.

Реже используется конструкция rgb(...), которая позволяет задавать красную, зеленую и синюю компоненты в десятичном или процентом виде:

```
A.content {color: rgb(0%,0%,0%)}
```

```
A.menu {color: rgb(51,0,170)}
```

URL

URL задаются конструкцией `url(...)`. Например, следующий CSS-код добавляет фоновое изображение для страницы:

```
BODY {background-image: url(images/bg.jpg);}
```

Шрифты

Шрифт – набор начертаний букв и знаков. В компьютере шрифт представляет собой файл, в котором описано, как должны отображаться на мониторе или принтере различные символы: буквы, цифры, знаки пунктуации и др. Часто шрифты содержат только начертания для латинского алфавита и не имеют, например, поддержки кириллицы. Существуют Unicode-шрифты, которые содержат символы для всех языков. Основные форматы файлов шрифтов: TTF – TrueType и его расширение OTF – OpenType.

Типы шрифтов:

serif – шрифты с засечками (антиквенные), например: Times New Roman, Georgia.

sans-serif – рубленые шрифты (шрифты без засечек или гротески), типичные представители – Arial, **Impact**, Tahoma, Verdana;

cursive – курсивные шрифты: **Comic Sans MS**;

fantasy – декоративные шрифты, например: *Curly M.I.*

monospace – моноширинные шрифты, ширина каждого символа одинакова. Примеры: Courier New, Lucida Console.

Засечками называют элементы на концах штрихов букв. Сравним букву шрифта Times New Roman и букву шрифта Arial.



Рисунок 3.1. Сравнение буквы М антиквенного и рубленого шрифта.

Пунктирными линиями обведены засечки.

Использование шрифтов с засечками облегчает чтение текста с бумаги, поэтому такие шрифты обычно используют для набора основного текста в книгах. Для web-сайтов основной текст чаще набирают шрифтом без засечек: Arial, Tahoma, Trebuchet MS, Verdana.

Текст

CSS позволяет управлять свойствами шрифта и текста.

font-family – задает начертание шрифта. Можно указать несколько значений через запятую. Браузер проверит первый шрифт из списка: если шрифт установлен на компьютере пользователя, то браузер применит его, если нет – перейдет ко второму шрифту и т.д. Последним в списке обычно указывается общий тип шрифта *serif*, *sans-serif*, *cursive*, *fantasy* или *monospace*. Пример:

```
font-family: Georgia, 'Times New Roman', serif
```

Если на компьютере пользователя установлен шрифт Georgia, то будет использоваться он, если нет – то Times New Roman. Если же и Times New Roman отсутствует, то браузер будет использовать шрифт с засечками, который установлен на компьютере.

Еще в CSS2 была реализована поддержка метода @font-face для загрузки недостающих шрифтов с сервера, но до недавнего времени не все браузеры поддерживали эту возможность. Сейчас @font-face работает в последних версиях FireFox, Opera, Safari. Internet Explorer реализует @font-face с 4 версии, но поддерживает только шрифты в формате EOT (Embedded OpenType), которые могут быть получены из TrueType и OpenType программой-конвертером.

font-size – размер шрифта. Может задаваться абсолютным значением в пунктах (pt) или пикселях (px) или относительным – в процентах (%) или в em. Пример:

```
font-size: 12pt
```

или

```
font-size: 150%
```

font-style – задает начертание текста: normal (обычное), italic (курсивное) или oblique (наклонное). Курсивное начертание является специальной измененной версией шрифта, имитирующей рукописный текст с наклоном вправо. Наклонное начертание получается из обычного наклоном букв. Различие видно на примере:

The five boxing wizards jump quickly.
The five boxing wizards jump quickly.
The five boxing wizards jump quickly.

Рисунок 3.2. Нормальное, курсивное и наклонное начертание.

Обычно браузер не может отобразить наклонное начертание и заменяет его курсивным.

font-weight – позволяет изменить уровень жирности текста: `normal` (обычная), `bold` (полужирная). Действие аналогично тегу ``.

В спецификации CSS 2.1 определены и другие значения свойства `font-weight` помимо `normal` и `bold`, но на данный момент браузеры плохо их поддерживают.

color – задает цвет текста (см. пункт «Цвета» этой лекции). Например, зададим красный цвет для всех заголовков:

```
H1, H2, H3, H4, H5, H6 {color: #ff0000}
```

или

```
H1, H2, H3, H4, H5, H6 {color: red}
```

line-height – межстрочный интервал (интерлиньяж), указывает расстояние между строками текста. Может задаваться числом как множитель от текущего размера шрифта, в процентах, а также в пунктах (pt), пикселях (px) и других единицах измерения CSS. Пример:

```
line-height: 1.5; /* полуторный интервал */
```



В программировании принято отделять целую часть числа от дробной точкой, как в английском языке. В русском языке используется запятая.

text-decoration – задает оформление текста. Варианты: `line-through` (перечеркнутый), `overline` (линия над текстом), `underline` (подчеркивание), `none` (отключение эффектов). Например, отключим подчеркивание у ссылок:

```
A {text-decoration: none}
```

text-align – выравнивание текста в блоке: `left` (по левому краю), `center` (по центру), `right` (по правому краю) или `justify` (по ширине). Пример:

```
P {text-align: justify}
```

text-indent – отступ первой строки («красная строка»). Длина отступа может задаваться в процентах (%) от ширины текстового блока, пикселях (px), пунктах (pt) и др. Пример:

```
P {text-indent: 1.25cm}
```

Свойства font-style, font-variant, font-weight, font-size, font-family и line-height можно задать в одном правиле:

```
font: font-style font-weight font-size/line-height font-family
```

Значения font-size и font-family являются обязательными, остальные можно не указывать, например:

```
H1 {font: bold 14pt/1.5 sans-serif}
```



Ресурсы в Интернете

- Единицы измерения. <http://www.htmlbook.ru/content/?id=60>
- Основы CSS. Текст: <http://htmlbook.ru/content/?id=55>
- Оформление текста с помощью CSS.
<http://www.intuit.ru/department/internet/operawebst/29/>
- Общие шрифты для всех версий Windows и их эквиваленты для Macintosh. <http://www.ampsoft.net/webdesign-1/WindowsMacFonts.html>
- Антиквенные шрифты*. <http://paratype.ru/help/class/default.asp?ClassCode=10000>
- Шрифт с засечками*. <http://www.webimg.ru/node/166>
- Читательность*. <http://designformasters.info/posts/readability/#serif>
- Как выбрать шрифт для веб-сайта*. <http://seleckis.lv/journal/shrifty/kak-vyibrat-shrift-dlya-web-sayta>
- Тенденции мировой типографики*. <http://habrahabr.ru/blogs/typography/67671/>
- CSS Font-Size: em vs. px vs. pt vs. percent*. <http://habrahabr.ru/blogs/webdev/42151/>
- @font-face или назад в будущее*. <http://lovtsevich.com/2009/10/26/font-face-ili-nazad-v-budushhee/>



Задания:

- а) Создайте новую страницу. Поместите на нее текст произвольного содержания. С помощью CSS задайте следующие параметры для заголовка: размер шрифта – 16 pt, полужирный, выравнивание по центру. Для текста – размер шрифта 12 pt, межстрочный интервал – полуторный, красная строка – 1,5 см. Подберите подходящий шрифт для заголовка и текста.
- б*) Добавьте на страницу пиктограммы с помощью шрифта Wingdings.

Лекция 3.3. CSS-свойства: поля, заполнение, границы

В CSS каждый элемент располагается в блоке, которому можно задать значения полей (*margin*), заполнения (*padding*) и границы (*border*). Поле является отступом элемента от соседних, а заполнение – пустой областью между границей и содержимым (см. рис. 3.3.).

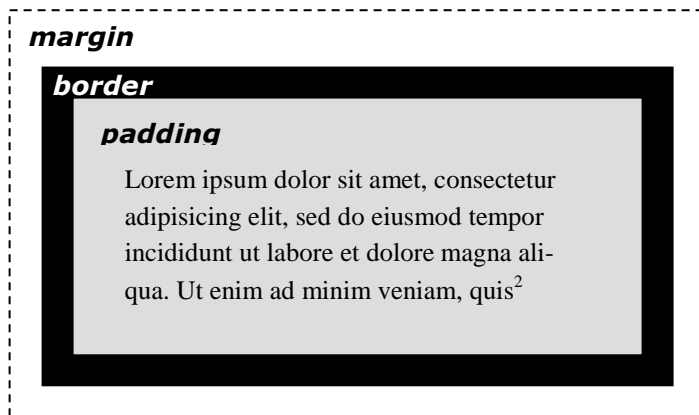


Рисунок 3.3 Бокс (*box*) элемента.

¹ Здесь и далее в примерах используется «рыба». Подробнее: «Ководство» § 67. Lorem ipsum: <http://www.artlebedev.ru/kovodstvo/sections/67/>

Ширина полей и заполнения задается следующими CSS свойствами:

margin-top, **margin-right**, **margin-bottom**, **margin-left** – для верхней, правой, нижней, левой стороны поля.

margin – сокращенная запись. Задает значение сразу для всех сторон. Пример:

```
P {margin: 10px}
аналогично записи
P {
margin-top: 10px;
margin-right: 10px;
margin-bottom: 10px;
margin-left: 10px;
}
```

Если для **margin** указать два значения через пробел, то первое из них будет задавать ширину верхнего и нижнего поля, а второе – левого и правого. Если указать три значения, то первое будет присваиваться верхнему полю, второе – левому и правое, а третье – нижнему. Наконец, при указании четырех значений, они поочередно будут указывать верхнее, правое, нижнее и левое поля.

padding-top, **padding-right**, **padding-bottom**, **padding-left** – устанавливают ширину заполнения¹ сверху, справа, снизу и слева от содержимого соответственно.

padding – устанавливает значение сразу для всех сторон.

Padding может принимать не только одно, но и 2, 3 или 4 значения. См. примечание для **margin**.

Для **margin** и **padding** можно задавать значение **auto**. В этом случае браузер сам автоматически рассчитает величину полей и заполнения.

¹ - В литературе встречаются различные переводы термина **padding**: заполнение, набивка, поле. Чтобы избежать путаницы, на занятиях рекомендуется использовать английское название.

Для границ можно задать толщину, цвет и стиль:

border-width – толщина границы;

border-color – цвет границы (по умолчанию – черный);

border-style – стиль границы. Может принимать значения `solid` (по умолчанию), `dotted`, `dashed`, `double`, `groove`, `ridge`, `inset` или `outset`. На рис. 3.4 представлены все виды границ, `border-width` установлен в 5 пикселей.

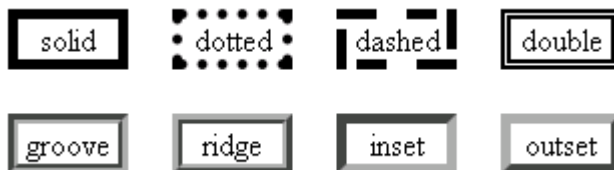


Рисунок 3.4. Виды границ.

Существует сокращенная запись: свойство **border** задает одновременно толщину, цвет и стиль. Значения указываются через пробел в любом порядке. Например:

```
<P style="border: solid 1px green">Текст</P>
```

Можно задавать стили отдельно для верхней, правой, нижней и левой границы, но это редко используется на практике. Например:

HTML-код:

```
<html>
<head>
<title>Пример</title>
<style>
н3 {
border-top: 2px dashed black;
border-bottom: 2px dashed black;
border-left: 0;
border-right: 0;
}
</style>
<body>
```

```
<h3>Заголовок</h3>
</body>
</html>
```

В браузере:

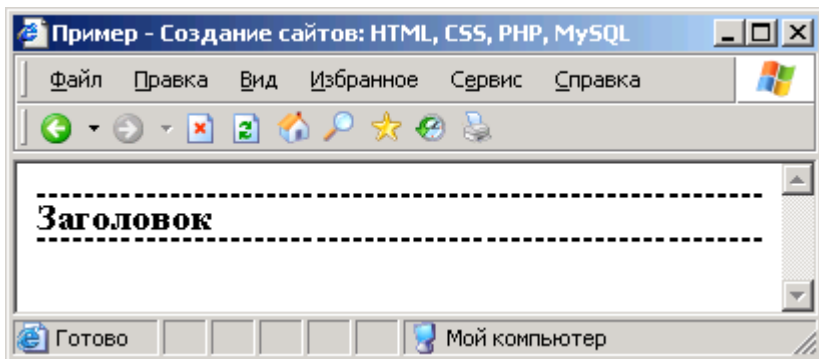


Рисунок 3.5 Задание свойств границ по отдельности

Возможно передавать в `border-width`, `border-color` и `border-style` не один, а до четырех параметров, как для `margin` и `padding`. Также существуют свойства для толщины, цвета и стиля каждой границы, например: `border-top-width`, `border-right-color`, `border-bottom-style` и др.

В предыдущем примере граница растянулась по всей ширине окна браузера. Это произошло потому, что многие HTML элементы по умолчанию занимают 100% ширины элемента, в которые они вложены. Для определения размера в CSS существуют свойства `width` и `height`. Чаще всего ширину и высоту задают в пикселях (px) или в процентах (%) от ширины родительского элемента. Рассмотрим пример:

HTML-код:

```
<html>
<head>
<title>Пример</title>
<style>
P {font-size: 10pt}
```

```
#text1 {
border: 1px solid black;
}

#text2 {
border: 1px solid black;
width: 300px;
}

#text3 {
border: 1px solid black;
width: 50%;
}
</style>
<body>
<p id="text1">Quo usque tandem abutere, Catilina, patientia nostra? quam diu etiam furor iste tuus nos eludet? quem ad finem sese effrenata iactabit audacia?</p>
<p id="text2">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
<p id="text3">Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.</p>
</body>
</html>
```

Размеры первого абзаца не указаны в стиле, ширина первого абзаца задана абсолютно в пикселях, а третьего – относительно ширины окна.

Если ширина или высота не заданы, они автоматически вычисляются браузером, исходя из размеров содержимого: для первого абзаца браузер установил ширину, равную ширине окна (100%). Во втором и третьем абзаце ширина задана, но не задана высота, поэтому браузер сам подобрал ее так, чтобы весь текст поместился в элемент.

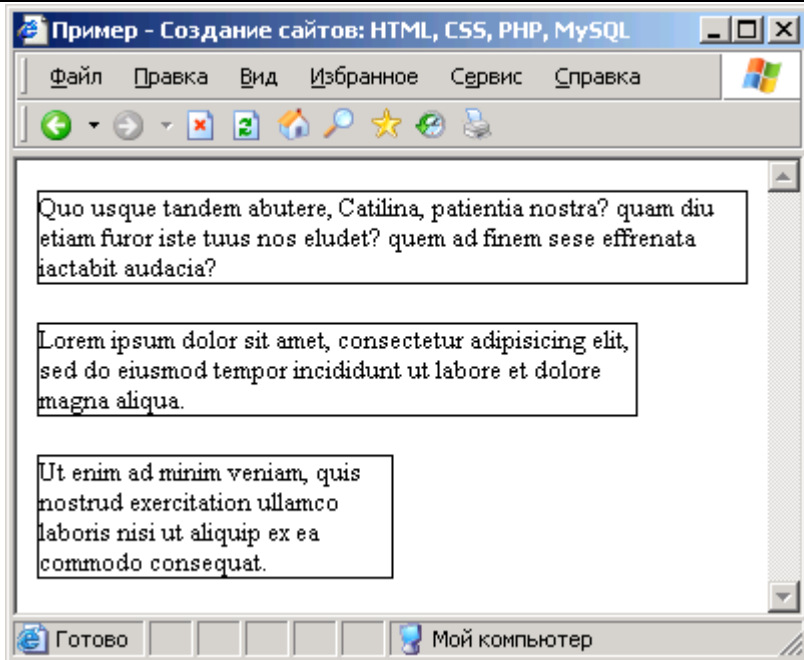


Рисунок 3.6. Отображение примера в браузере

Теперь, если пользователь изменит размер окна, пропорционально изменится ширина тех элементов, где она была задана в процентах.

Уменьшим размер окна браузера. У первого и третьего абзаца уменьшится ширина, а высота увеличится, чтобы вместить весь текст. Размеры второго абзаца останутся неизменными, появятся полосы прокрутки.

Поведение браузеров различается, если для элемента заданы и ширина, и высота, а содержимое не вмещается в эти размеры. Internet Explorer увеличит размеры элемента. Браузеры, полностью поддерживающие стандарт CSS, такие как Firefox, отобразят содержимое поверх блока.

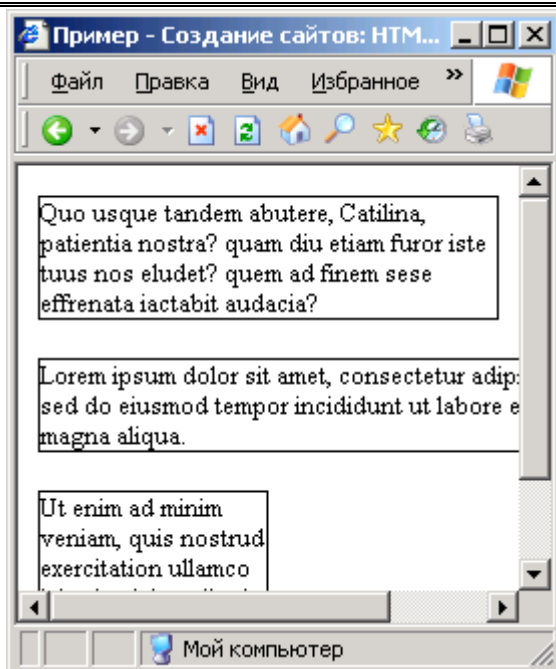


Рисунок 3.7. Отображение примера в браузере при уменьшении ширины окна

Можно задавать минимальные и максимальные размеры свойствами `min-width`, `min-height` и `max-width`, `max-height`. К сожалению, эти свойства не поддерживает браузер Internet Explorer версии 6 и ниже. Пока этим браузером пользуется значительная часть пользователей, указывать минимально и максимально допустимые размеры не рекомендуется, т.к. это может привести к ошибкам отображения в IE6.

Общие размеры элемента складываются так:

Ширина = `width` + `padding` + `border` + `margin`

Высота = `height` + `padding` + `border` + `margin`

Т.е. `width` и `height` задают только размеры содержимого, не включая поля, заполнение и границу! См. рис. 3.8.

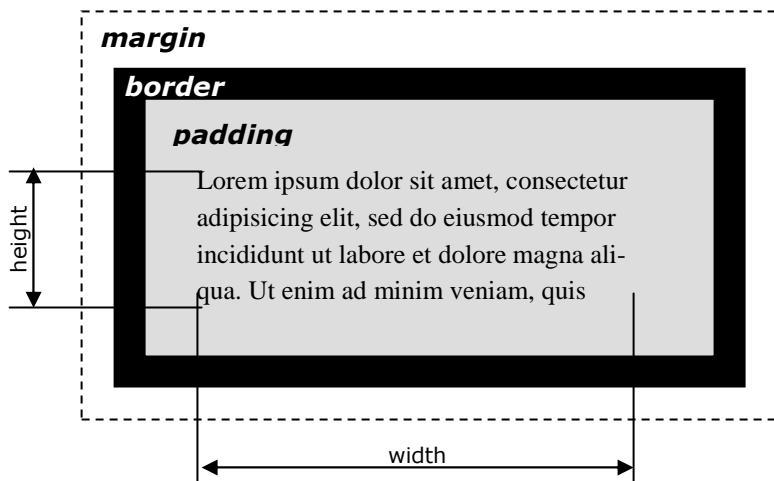


Рисунок 3.8. Бокс элемента и размеры содержимого



Ресурсы в Интернете

- Модель компоновки CSS – боксы, границы, поля, заполнение. <http://www.intuit.ru/department/internet/operawebst/30/>
- Справочник [htmlbook.ru](http://htmlbook.ru/css/). Границы, поля, отступы. <http://htmlbook.ru/css/>



Задания:

- Реализуйте пример с рис. 3.6. Текст абзацев вставьте свой. Отключите поля и заполнение для всех элементов на странице (*). Проанализируйте результат. Добавьте заголовок тегом H1. Установите поля и отступы для тегов BODY, H1 и P. Для каждого абзаца установите разный вид и толщину границ.
- Создайте параграф размерами 300×100 пикселей. Поместите туда большой текст. Сравните поведение страницы в Internet Explorer и Firefox.
- Задайте для второго параграфа отрицательное верхнее поле. Оцените результат.

Лекция 3.4. CSS-свойства: фон, оформление таблиц

Фон

Как и в языке HTML, в CSS фоном служит заливка цветом или изображение. Фоновое изображение может быть повторяющимся.

background-color – устанавливает цвет фона. Пример:

```
TD.head {background-color: #ffff00}
```

background-image – устанавливает в качестве фона изображение:

```
BODY {background-image: url(images/bg.jpg)}
```

background-attachment – задает поведение фонового изображения при прокрутке. По умолчанию задается значение `scroll` – фон прокручивается вместе с содержимым. Значение `fixed` делает фон неподвижным.

background-position – начальное положение фонового изображения по горизонтали (`left`, `center`, `right`) и вертикали (`top`, `center`, `bottom`). Вместо ключевых слов можно указывать расстояние в пикселях или процентах.

background-repeat – указывает, в каком направлении должно размножаться фоновое изображение:

`repeat` – по горизонтали и вертикали (по умолчанию);

`repeat-x` – только по горизонтали;

`repeat-y` – только по вертикали;

`no-repeat` – отключить повторение.

Пример:

Используя изображение одного вагона, составим в фоне поезд.

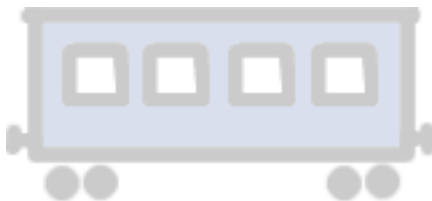


Рисунок 3.9. Фоновое изображение.

CSS код:

```
BODY {  
background-image: url('coach.png');  
background-repeat: repeat-x;  
background-position: 80px 100px;  
}
```

В браузере:

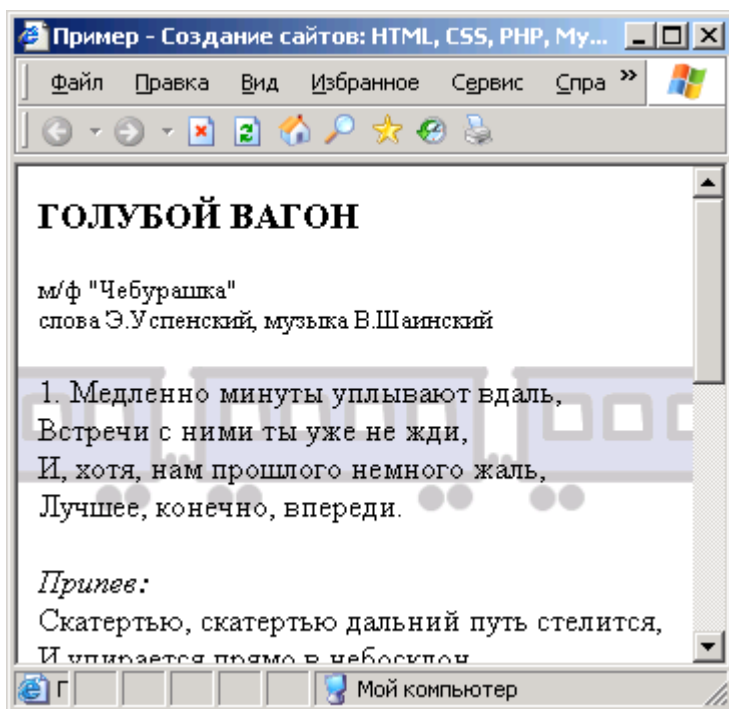


Рисунок 3.10. Фоновое изображение на странице.

Таблицы

Свойства CSS могут применяться к таблицам, их строкам и ячейкам для задания свойств текста и шрифта, управления фоном, полями, границами, размерами и т.п.

Создадим таблицу и применим к ней CSS-стили. В таблицу внесем данные о популярности различных браузеров¹. Для заголовка таблицы используем тег <th>...</th>.

HTML-код:

```
<html>
<head>
<title>Популярность браузеров в мире</title>
</head>
<body>
<table>
  <tr>
    <th>Год\Браузер</th>
    <th>IE</th>
    <th>Firefox</th>
    <th>Safari</th>
    <th>Opera</th>
  </tr>
  <tr>
    <td>2010</td>
    <td>61.43%</td>
    <td>24.40%</td>
    <td>4.55%</td>
    <td>2.37%</td>
  </tr>
  <tr>
    <td>2009</td>
    <td>69.13%</td>
    <td>22.67%</td>
    <td>3.58%</td>
    <td>2.18%</td>
  </tr>
```

¹- По данным Net Applications для первого квартала года.

```

<tr>
  <td>2008</td>
  <td>77.83%</td>
  <td>16.86%</td>
  <td>2.65%</td>
  <td>1.84%</td>
</tr>
<tr>
  <td>2007</td>
  <td>79.38%</td>
  <td>14.35%</td>
  <td>4.70%</td>
  <td>0.50%</td>
</tr>
</table>
</body>
</html>

```

Без CSS-оформления таблица будет выглядеть так:

Год\Браузер	IE	Firefox	Safari	Opera
2010	61.43%	24.40%	4.55%	2.37%
2009	69.13%	22.67%	3.58%	2.18%
2008	77.83%	16.86%	2.65%	1.84%
2007	79.38%	14.35%	4.70%	0.50%

Рисунок 3.11. Отображение таблицы по умолчанию

По умолчанию содержимое заголовочных ячеек отображается жирным шрифтом с выравниванием по центру.

Добавим в тег `<head>...</head>` тег `<style>...</style>`, а к тегу `<table>...</table>` атрибут `id="browser_stats"`. Запишем CSS-правила для таблицы. Для заголовочных ячеек установим серый фон и отступ содержимого от границ (`padding`) в половину высоты строки, для ячеек с данными – выравнивание по правому краю и `padding` три десятых от высоты строки. Вокруг таблицы зададим двойную рамку, а для ячеек – обычную одинарную.

Код:

```
<style>
/* стиль таблицы */
TABLE#browser_stats {
  border: 3px double black;
}
/* стиль заголовочных ячеек */
TABLE#browser_stats TH{
  border: 1px solid black;
  background-color: gray;
  padding: 0.5em;
}
/* стиль ячеек с данными */
TABLE#browser_stats TD{
  border: 1px solid black;
  padding: 0.3em;
  text-align: right;
}
</style>
```

В браузере:

Год\Браузер	IE	Firefox	Safari	Опера
2010	61.43%	24.40%	4.55%	2.37%
2009	69.13%	22.67%	3.58%	2.18%
2008	77.83%	16.86%	2.65%	1.84%
2007	79.38%	14.35%	4.70%	0.50%

Рисунок 3.12. Отображение таблицы с заданными CSS-стилями

Виден существенный недостаток: у каждой ячейки появилась собственная рамка. Чтобы этого не происходило, необходимо указать в правилах для таблицы свойство `border-collapse` со значением `collapse`. Результат:

Год\Браузер	IE	Firefox	Safari	Opera
2010	61.43%	24.40%	4.55%	2.37%
2009	69.13%	22.67%	3.58%	2.18%
2008	77.83%	16.86%	2.65%	1.84%
2007	79.38%	14.35%	4.70%	0.50%

Рисунок 3.13. Эффект слияния границ соседних ячеек

Теперь применим к той же таблице другое форматирование. Разделим таблицу двумя линиями на 3 части: названия браузеров, годы и процентные данные. Названия браузеров и процентные доли выровняем по центру, годы – по правому краю. Зададим одинаковую ширину для столбцов с информацией по браузерам.

Год\Браузер	IE	Firefox	Safari	Opera
2010	61.43%	24.40%	4.55%	2.37%
2009	69.13%	22.67%	3.58%	2.18%
2008	77.83%	16.86%	2.65%	1.84%
2007	79.38%	14.35%	4.70%	0.50%

Рисунок 3.14. Оформление таблицы с двумя разделительными линиями

Чтобы применить правила CSS к левой колонке (годы), нам придется задать новый класс `lc` и прописать атрибут `class="lc"` во все ячейки левой колонки.

Горизонтальная линия создается путем указания свойства `border-bottom` для ячеек `TH`, вертикальная – `border-left` для ячеек класса `lc`.

Код-страницы:

```
<html>
<head>
<title>Популярность браузеров в мире</title>
<style>
TABLE#browser_stats {
border-collapse: collapse;
}

TABLE#browser_stats TH{
border-bottom: 1px solid black;
}

TABLE#browser_stats TD{
padding: 0.3em;
text-align: center;
width: 70px;
}

TABLE#browser_stats .lc{
text-align: right;
border-right: 1px solid black;
width: 100px;
}
</style>
</head>
<body>
<table id="browser_stats">
  <tr>
    <th class="lc">Год\Браузер</th>
    <th>IE</th>
    <th>Firefox</th>
    <th>Safari</th>
    <th>Opera</th>
  </tr>
  <tr>
    <td class="lc">2010</td>
```

```
<td>61.43%</td>
<td>24.40%</td>
<td>4.55%</td>
<td>2.37%</td>
</tr>
<tr>
<td class="lc">2009</td>
<td>69.13%</td>
<td>22.67%</td>
<td>3.58%</td>
<td>2.18%</td>
</tr>
<tr>
<td class="lc">2008</td>
<td>77.83%</td>
<td>16.86%</td>
<td>2.65%</td>
<td>1.84%</td>
</tr>
<tr>
<td class="lc">2007</td>
<td>79.38%</td>
<td>14.35%</td>
<td>4.70%</td>
<td>0.50%</td>
</tr>
</table>
</body>
</html>
```



Ресурсы в Интернете

- Оформление таблиц. <http://htmlbook.ru/content/?pid=13>
- Тег TH. <http://htmlbook.ru/html/th.html>
- Стилиевое свойство border-collapse. <http://htmlbook.ru/css/border-collapse.html>
- Оформление таблиц.
<http://www.intuit.ru/department/internet/operawebst/33/>



Задания:

- а) Реализуйте примеры таблиц из лекции (рис. 3.13. и рис. 3.14.).
 б) Измените форматирование таблицы на следующий вариант:

Год\Браузер	IE	Firefox	Safari	Opera
2010	61.43%	24.40%	4.55%	2.37%
2009	69.13%	22.67%	3.58%	2.18%
2008	77.83%	16.86%	2.65%	1.84%
2007	79.38%	14.35%	4.70%	0.50%

Рисунок. 3.15. Задание 3.3.б.

- в) Измените форматирование: задайте одинаковую высоту строк и установите чередование фона («зебра»).

У к а з а н и е: используйте разные классы стилей для четных и нечетных строк таблицы.

Год\Браузер	IE	Firefox	Safari	Opera
2010	61.43%	24.40%	4.55%	2.37%
2009	69.13%	22.67%	3.58%	2.18%
2008	77.83%	16.86%	2.65%	1.84%
2007	79.38%	14.35%	4.70%	0.50%

Рисунок. 3.16. Задание 3.3.в.

г*) Модифицируйте таблицу из задания «в». Чередование цвета фона у строк замените чередованием цвета фона у колонок.
д*) Отключите фон у ячеек таблицы. Добавьте фоновое изображение для таблицы.

Лекция 3.5. Теги DIV и SPAN, псевдоклассы

Теги DIV и SPAN

До сих пор в лекциях мы применяли стили CSS к тегам, уже имеющим заранее заданную функцию: таблицам, заголовкам, параграфам и т.д. Но иногда нужно применить стили к фрагменту содержимого, не включенного в отдельный тег. Например, выделить фоном несколько слов в тексте.

Теги `<div>...</div>` и `...` используются там, где не подходит никакой другой тег. Сами по себе они не определяют никакого форматирования, но удобны для привязки к ним стилей. При этом DIV является блочным элементом, а SPAN – строчным.

Основное различие между блочными и строчными элементами заключается в следующем: строчные элементы идут друг за другом в строке текста, а блочные – располагаются один по другим. К строчным элементам относятся такие теги, как `<a>`, ``, `<input>`, `<select>`, ``, `<sub>`, `<sup>` и др. К блочным: `<div>`, `<form>`, `<h1>...<h6>`, ``, `<p>`, `<table>`, `` и некоторые другие. Рассмотрим различие на примере. Для тега `` указано стилевое правило, задающее цвет фона.

HTML-код:

```
<span style="background-color: #eeeeee">Строчные элемен-  
ты</span>  
<sub>располагаются</sub>  
  
<sup>и идут друг за другом.</sup>
```

В браузере:

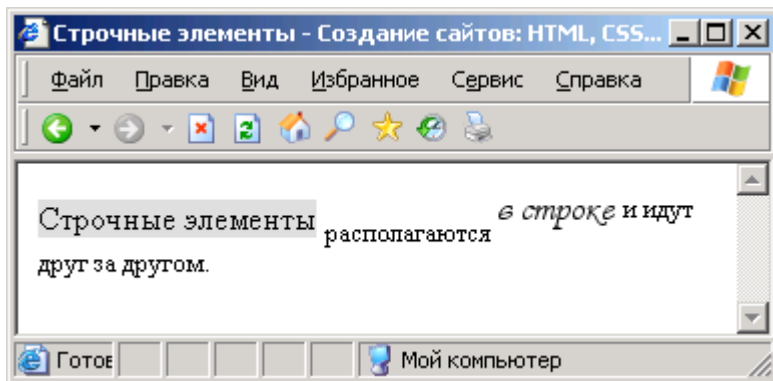


Рисунок 3.17. Поведение строчных элементов.

Рассмотрим пример для блочных тегов:

```
<html>
<head>
<title>Блочные элементы</title>
<style>
h3, DIV, TABLE {
border: black dotted 1px;
margin: 5px;
padding: 5px;
}
</style>
</head>
<body>
<h3>Заголовок</h3>
<div>Содержимое &lt;div&gt; №1</div>
  <div>Вложенный &lt;div&gt; №1</div>
  <div>Вложенный &lt;div&gt; №2</div>
</div>
<table>
<tr><td>Таблица из одной ячейки</td></tr>
</table>
</body>
</html>
```

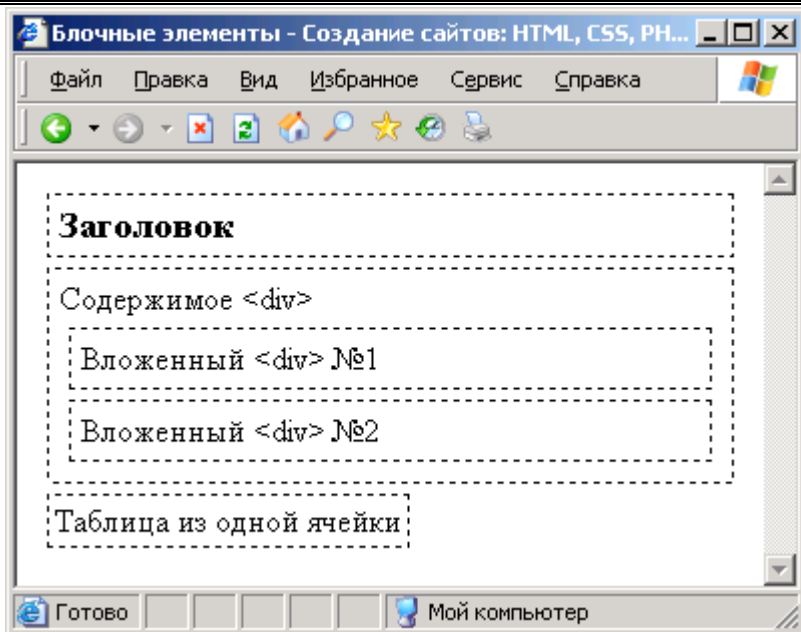


Рисунок 3.18. Поведение блочных элементов.

Блочные элементы располагаются друг под другом, многие занимают всю возможную ширину. Блочные элементы могут включать в себя строчные и другие блочные. Но строчные элементы не могут содержать блочные!

Еще одним отличием является то, что для строчных элементов не работают такие свойства, как `margin-top`, `margin-bottom`, `padding-top` и `padding-bottom`. Исключением являются теги ``, `<input>`, `<textarea>` и `<select>` – для них можно задавать отступы `padding-top` и `padding-bottom`.

Псевдоклассы

В лекции 3.1. мы рассмотрели способы привязки правил оформления CSS к элементам документа HTML: по названию тега, по имени класса, по ID и т.п. В CSS также существует несколько *псевдоклассов*. С помощью псевдоклассов можно задать стиль в зависимости от состояния элемента или его положения в документе.

Для ссылок определено 4 псевдокласса:

link – ссылки, которые не посещались пользователем;

visited – посещенные ссылки;

active – активная (нажатая) ссылка;

hover – ссылка, на которую наведен курсор.

Пример:

```
<html>
<head>
<title>Пример</title>
<style>
A:link, A:visited {
color: black;
font-family: Verdana, sans-serif;
text-decoration: none;
}

A:hover {
color: #de7300;
text-decoration: underline;
}
</style>
</head>
<body>
<a href="index.html">Главная</a><br>
<a href="hobby.html">Мое хобби</a><br>
<a href="photo.html">Фотоальбом</a><br>
</body>
</html>
```

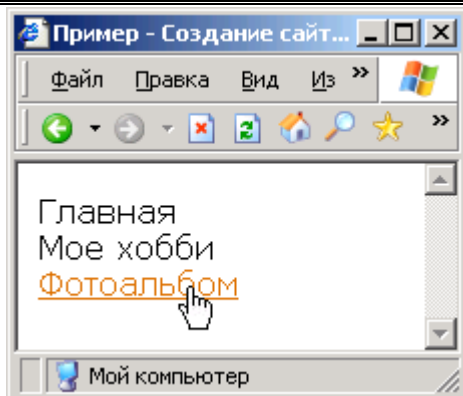


Рисунок 3.19. Пример: меню сайта

Internet Explorer версии 6 поддерживает свойства `hover` и `active` только для ссылок, тогда как более современные браузеры (Firefox, IE версии 7 и выше и другие) могут применять эти свойства и к другим элементам страницы, например к ячейкам таблицы.

Псевдокласс (или псевдоэлемент) **`first-line`** – применяется для блочных элементов. Задаёт форматирование первой строки текста. Пример:

```
<html>
<head>
<title>Пример</title>
<style>
P:first-line {text-decoration: underline}
</style>
</head>
<body>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua. Ut enim ad minim veniam, quis nos-
trud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. </p>
</body>
</html>
```

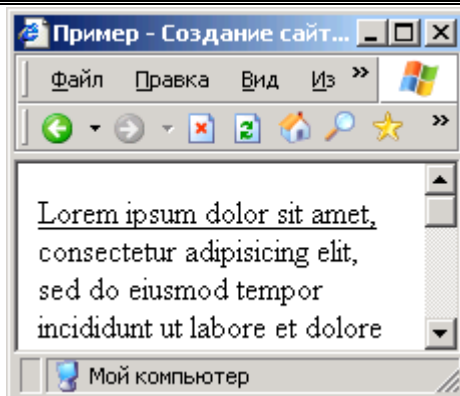


Рисунок 3.20. Пример использования псевдокласса *first-line*.

Псевдокласс (или псевдоэлемент) **first-letter** – позволяет задать форматирование первой буквы текста. Для примера создадим «буквицу» – начальную букву текста увеличенного размера:

```
<html>
<head>
<title>Пример</title>
<style>
P:first-letter {
color: red;
font-size: 200%;
border: red solid 1px;
padding: 2px;
margin: 2px;
}
</style>
</head>
<body>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing
elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua. Ut enim ad minim veniam, quis nos-
trud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. </p>
</body>
</html>
```

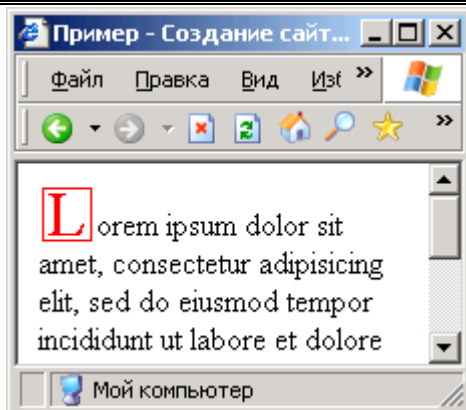


Рисунок 3.21. Использование псевдокласса *first-letter* для создания буквицы.



Ресурсы в Интернете

- Базовые контейнеры – элементы `div` и `span`.
<http://www.intuit.ru/department/internet/operawebst/22/2.html>
- Селекторы – псевдоэлементы и псевдоклассы.
<http://www.intuit.ru/department/internet/css2/5/6.html>
- Псевдоклассы. <http://stepbystep.htmlbook.ru/?id=58>



Задания:

а) На основе данных о популярности браузеров за 2010 год (из предыдущей лекции) создайте столбчатую диаграмму (гистограмму). Пример показан на рис. 3.22.

У к а з а н и е: используйте элементы `DIV` заданной ширины.

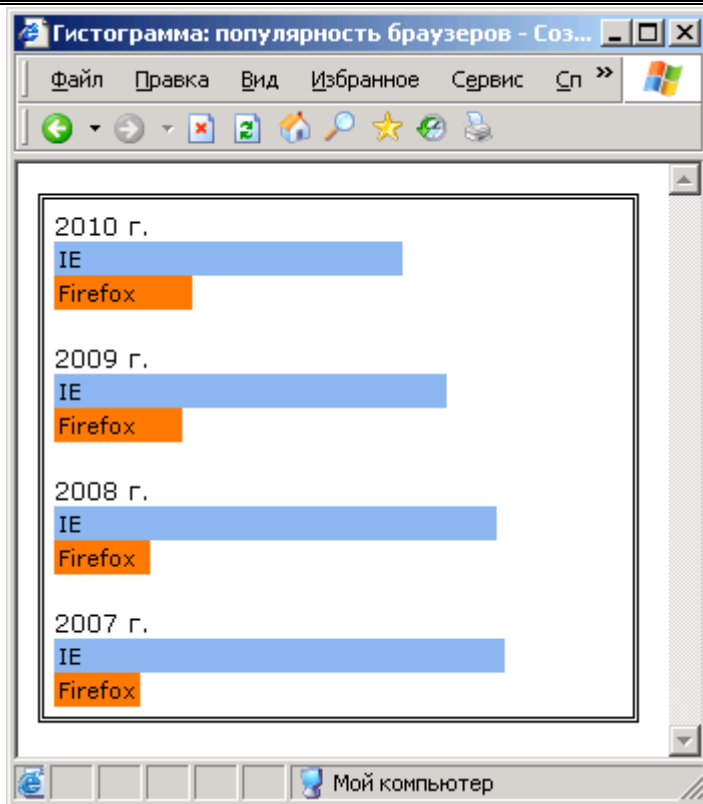


Рисунок. 3.22. Задание 3.4.a.

- б) С помощью псевдокласса first-letter создайте свой вариант буквицы. Подберите шрифт, размер, цвет и оформление
- в) Добавьте эффект выделения ссылок при наведении курсора на своем сайте.
- г*) Модифицируйте пример с рис. 3.17. С помощью таблицы реализуйте меню, как показано на рис. 3.23. Используя псевдокласс hover для строки таблицы, добавьте выделение пункта меню цветом при наведении курсора. Пример не будет работать в браузере Internet Explorer версии 6 и ниже.

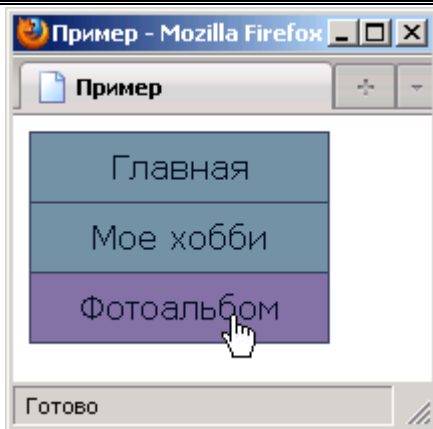


Рисунок 3.23. Задание 3.4.г.

Лекция 3.6. CSS-свойства: позиционирование

Установка координат элемента

С помощью CSS можно точно задать положение элемента на странице. Режимом позиционирования управляет свойство `position`:

`position` – устанавливает, каким образом вычисляется положение элемента в плоскости экрана. Существует четыре режима.

`position: static` – режим по умолчанию, элементы отображаются как обычно – в порядке следования в коде по правилам HTML.

`position: relative` – задает относительное свободное позиционирование. Значения атрибутов `top`, `right`, `bottom`, и `left` при этом задают смещение координат элемента страницы от точки, в которой он был отображен. Например, создадим CSS-замену тегу `^{...}`.

HTML-код:

```
<span style="font-size: 30pt">  
2<span style="font-size: 50%; position: relative; top: -  
1em;">8</span> = 256  
</span>
```

Чтобы поместить цифру «8» в верхний индекс, уменьшаем ее размер в половину и сдвигаем вверх на высоту строки (1 em). Свойство `top` указывает расстояние от первоначального положения относительно верхней границы документа. Для того чтобы поднять «8» наверх, мы указываем отрицательное значение `top`. В этом примере можно вместо свойства `top: -1em` написать `bottom: 1em`.

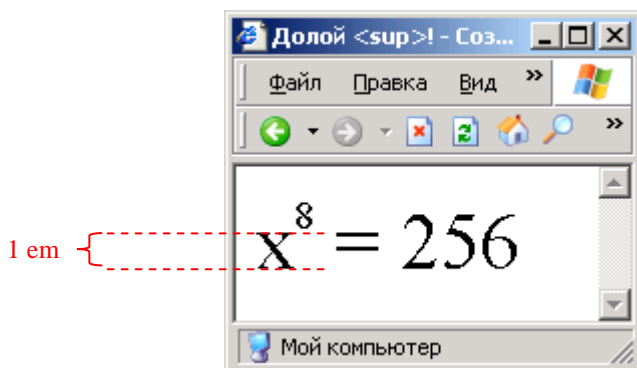


Рисунок 3.24. Замена тега `<sup>` средствами CSS.

При разработке сайтов таким способом пользоваться не рекомендуется. Для преобразования в верхний индекс лучше использовать специально предназначенный атрибут `vertical-align` со значением `sub` для нижнего индекса или `super` для верхнего

position: absolute – задает абсолютное свободное позиционирование. Значения атрибутов `top`, `right`, `bottom` и `left` и при этом задают абсолютные координаты элемента страницы относительно родителя. Создадим два контейнера `DIV` и воспользуемся `position: absolute` для указания их координат.

```
<html>
<head>
<title>Position: absolute</title>
<style>
DIV {
width: 100px;
height: 100px;
border: 3px double black;
padding: 5px;
position: absolute;
}

DIV#first {
background-color: #c0dcc0;
top: 40px;
left: 40px;
}

DIV#second {
background-color: #c0c0dc;
top: 80px;
left: 100px;
}

</style>
</head>
<body>
<div id="first">1</div>
<div id="second">2</div>
</body>
</html>
```

Для блоков задается отступ от верхнего и левого края свойствами `top` и `left`. Так как второй блок объявлен в HTML-коде позже, он перекрывает первый блок на странице.

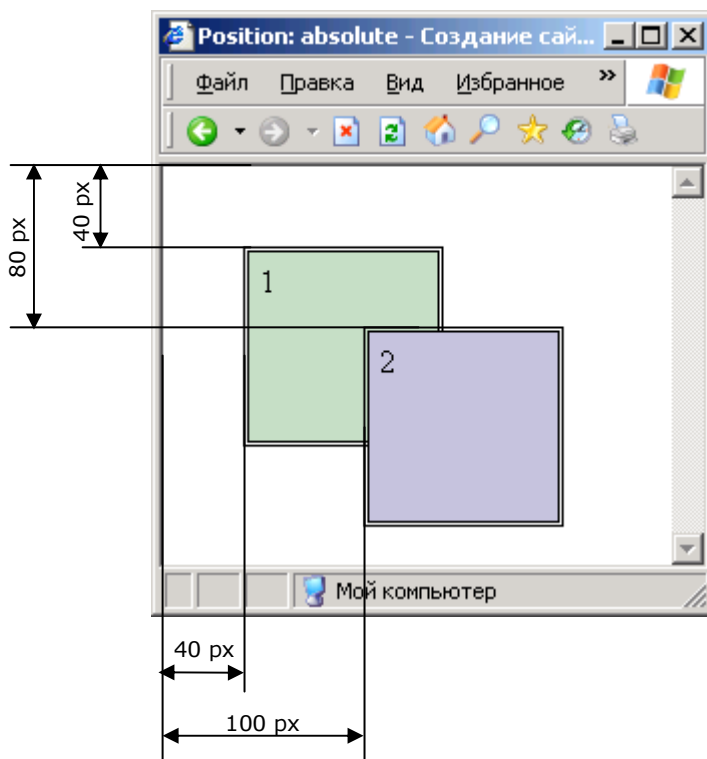


Рисунок 3.25. Использование абсолютного позиционирования.

Для управления порядком наложения элементов друг на друга необходимо использовать свойство **z-index**. Значением **z-index** является положительное или отрицательное число, задающее «высоту», на которой расположен элемент. Элементы с большим **z-index** накладываются сверху элементов с меньшим **z-index**. Чтобы в предыдущем примере первый блок оказался выше второго, необходимо для первого блока задать **z-index**, к примеру, равным двум, а для второго – единице.

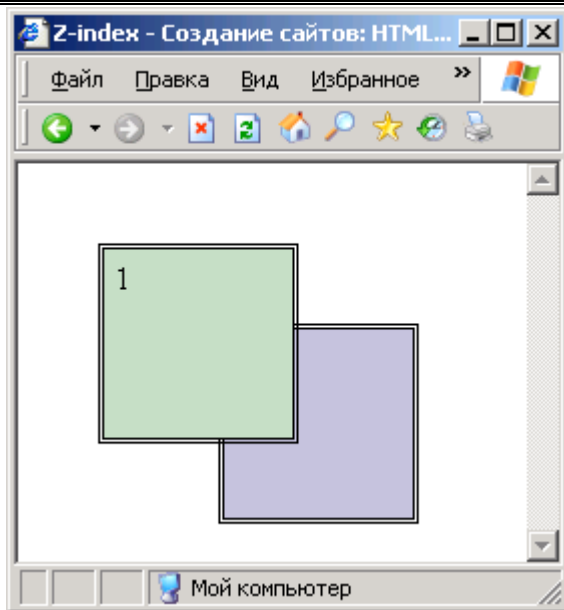


Рисунок 3.26. Использование z-index для изменения порядка наложения элементов

position: fixed – фиксирует элемент относительно окна. Элемент остается на месте даже при прокрутке страницы. К сожалению, режим `fixed` не работает в браузере Internet Explorer версии 6 и ниже, поэтому пока применять его не рекомендуется.

Плавающие элементы

В прошлой лекции мы узнали, что по умолчанию блочные элементы идут строго друг под другом. Изменить этот порядок можно сделав элементы «плавающими». Для этого служит CSS атрибут `float`. Он задает, по какой стороне будет выравниваться элемент: левой (`left`) или правой (`right`). Плавающий элемент будет стремиться к левой или правой стороне родительского элемента, а с других сторон он может обтекаться текстом или другими элементами.

При этом нужно помнить, что свойство `float` не работает одновременно с заданием позиционирования, рассмотренным в первой части лекции.

Наглядно работа `float` видна на примере:

```
<html>
<head>
<title>Плавающие элементы</title>
<style>
DIV#floating {
float: left;
}
</style>
</head>
<body>
Per Apollinem medicum et Aesculapium, Hygiamque et Pana-
ceam juro, deos deasque omnes testes citans, mepte viri-
bus et iudicio meo hos jusjurandum et hanc stipulationem
plene praestaturum.<br>
<div id="floating"></div>
Illum nempe parentum meorum loco habiturum spondeo, qui
me artem istam docuit, eique alimenta impertirurum, et
quibuscunque opus habuerit, suppeditaturum.<br>
Victus etiam rationem pro virili et ingenio meo aegris
salutarem praescripturum a pemiciosa vero et improba
eisdem prohibiturum. Nullius praeterea precibus
adductus, mortiferum medicamentum cuique propinabo,
neque huius rei consilium dabo. Caste et sancte colam et
artem meam.<br>
</body>
</html>
```

Контейнер `DIV` с изображением стремится к левому краю документа, а с остальных трех сторон он обтекается текстом (рис. 3.27.).



Рисунок 3.27. Обтекание текстом блочного элемента

Создадим пример с несколькими плавающими блоками. Зададим основной контейнер с фиксированной шириной, а в него поместим пять плавающих блоков с выравниванием по левому краю.

```
<html>
<head>
<title>Плавающие элементы</title>
<style>
DIV#main {
border: double black 3px;
width: 150px;
padding: 5px;
}
```

```
DIV.lefty {  
border: dashed black 1px;  
width: 30px;  
height: 30px;  
float: left;  
margin: 5px;  
text-align: center;  
}  
</style>  
</head>  
<body>  
<div id="main">  
<div class="lefty">1</div>  
<div class="lefty">2</div>  
<div class="lefty">3</div>  
<div class="lefty">4</div>  
<div class="lefty">5</div>  
</div>  
</body>  
</html>
```

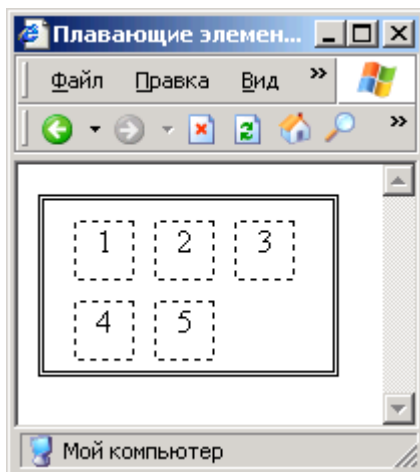


Рисунок 3.28. Пример: несколько плавающих блоков.

Первый блок выравнивается по левому краю родительского контейнера. Второй блок тоже стремится к левому краю, но так как место уже занято первым блоком, второй блок становится (обтекает) справа от первого. Аналогично

поступает третий блок. Четвертый блок уже не может встать справа от третьего, поэтому он помещается ниже остальных и выравнивается по левому краю. И наконец, пятый блок обтекает четвертый справа.

Можно одновременно использовать блоки с выравниванием по левому и правому краю.

```
<div style="float: left">&larr; налево</div>
```

```
<div style="float: right">направо &rarr;</div>
```

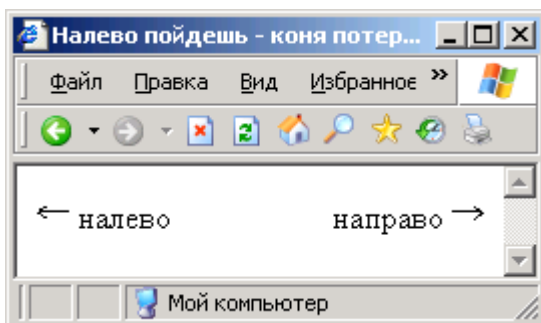


Рисунок 3.29. Пример: блоки с выравниванием по разным краям.

Еще одним свойством, связанным с плавающими элементами, является **clear**. Clear запрещает обтекание элемента с левой (*left*), правой (*right*) или с обеих сторон (*both*). По умолчанию значение – *none* – обтекание разрешено. Рассмотрим пример:

```
<html>
<head>
<title>Clear</title>
<style>
DIV {
border: solid black 1px;
width: 75px;
}

DIV.floating {
float: left;
}
```

```
</style>
</head>
<body>
<div class="floating">Блок 1</div>
<div class="floating">Блок 2</div>
<div style="clear: both">Блок с запретом обтекания</div>
<div class="floating">Блок 3</div>
<div class="floating">Блок 4</div>
<div class="floating">Блок 5</div>
</body>
</html>
```

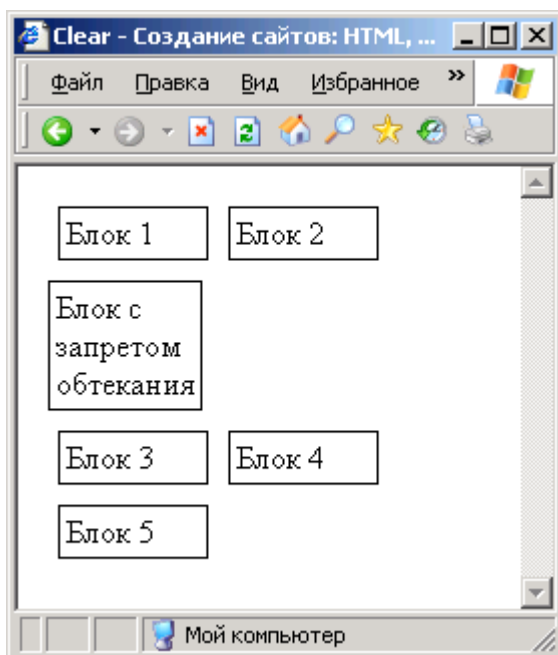


Рисунок 3.30. Использование правила *clear*.

При создании сайтов плавающие элементы, свойства `float` и `clear` часто используются для создания «каркаса» страниц сайта. Более подробно этот вопрос будет рассмотрен в следующей теме.



Ресурсы в Интернете

- Плавающие элементы и очистка.
<http://www.intuit.ru/department/internet/operawebst/35/>
- Статическое и относительное позиционирование CSS.
<http://www.intuit.ru/department/internet/operawebst/36/>
- Абсолютное и фиксированное позиционирование CSS.
<http://www.intuit.ru/department/internet/operawebst/37/>
- CSS Float в теории и на практике. <http://www.tulip.net/verstka/105-css-float-v-teorii-i-na-praktike.html>



Задания:

а) Используя плавающий блок, создайте буквицу, смещенную на одну строку вниз, как показано на рис. 3.31.

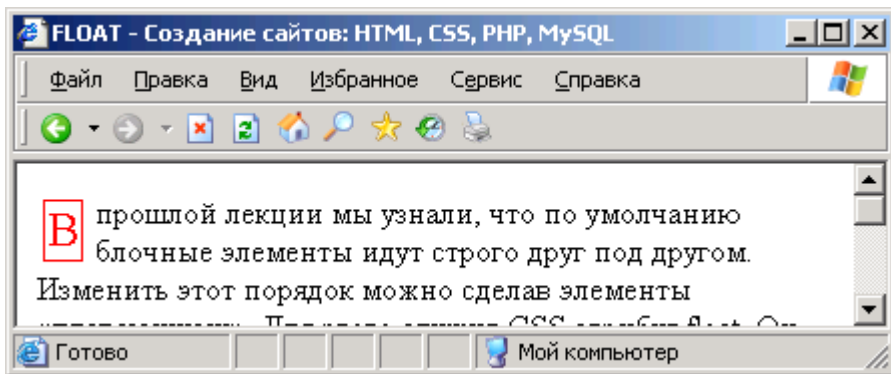


Рисунок 3.31. Задание 3.6.а.

б) С помощью относительного позиционирования каждой буквы создайте «эффект морской волны».

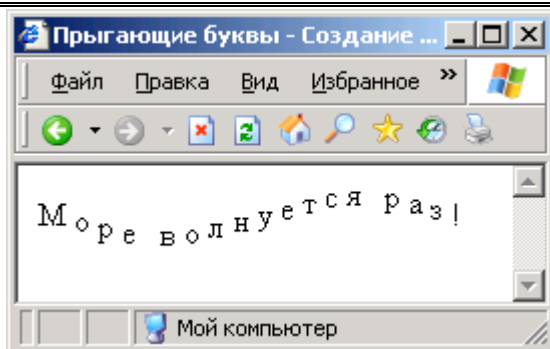


Рисунок 3.32. Задание 3.6.б.

в*) Посредством абсолютного позиционирования изобразите точечный график математической функции. Пример для $y=x^2$ показан на рис. 3.33.

У к а з а н и е: Создайте изображение с нарисованными на нем осями X и Y. Поверх него поместите точки графика абсолютным позиционированием.



Рисунок 3.33. Задание 3.6.в.

Лекция 4.1. Основы верстки. Табличная верстка.

Основы верстки

Под версткой веб-страницы понимают процесс ее создания путем компоновки текстовых и графических элементов. При создании дизайна сайта дизайнер разрабатывает макет в графическом редакторе. Макет является обычным изображением. Поэтому, чтобы использовать дизайн на сайте, верстальщик «превращает» макет в веб-страницу, где расположение элементов задается с использованием HTML и CSS.

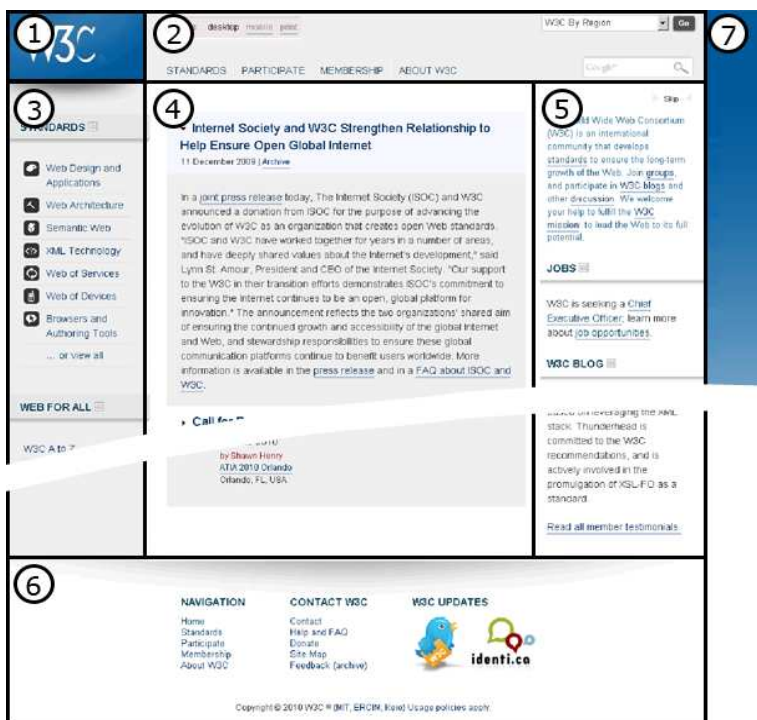


Рисунок 4.1. Основные блоки страницы¹ сайта <http://w3.org>

Как правило, веб-страница представляется как набор прямоугольных блоков.

¹ - Скриншот сайта сокращен по высоте, вырезана средняя часть.

В качестве примера рассмотрим сайт консорциума W3C (рис. 4.1.). Основные блоки на странице: 1 – логотип, 2 – верхняя часть (header), 3 – левая колонка, 4 – центральная колонка, 5 – правая колонка, 6 – нижняя часть (footer), 7 – общий фон страницы. Блоки в свою очередь могут содержать в себе другие более мелкие блоки: пункты меню, панели, и т.п. В верхней части располагается основное навигационное меню и форма поиска, в левой колонке – меню раздела, в правой колонке – важные объявления, в центральной – основное содержание страницы, а в нижней части дублируется верхнее меню (чтобы для перехода в другой раздел не нужно было прокручивать всю страницу снизу вверх) и контакты. Такая верстка страницы называется трехколоночной. Наиболее популярными являются трехколоночная и двухколоночная верстки. В двухколоночной, как правило, отсутствует правая колонка. Разумеется, существуют и другие варианты компоновки элементов, например в одну колонку. На рис. 4.2. показаны модели одно- и двухколоночных страниц.

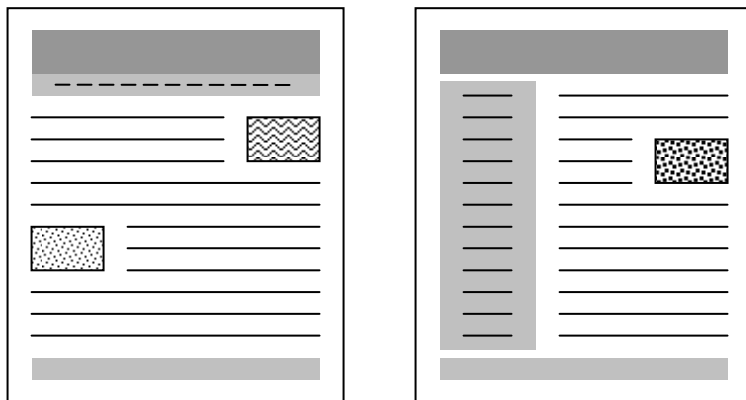


Рисунок 4.2. Одноколоночная и двухколоночная страницы.

Пример одноколоночной страницы – поисковая выдача Google.com, двухколоночной – страницы сайта президента России kremlin.ru, блога habrahabr.ru, gismeteo.ru и другие.

Фиксированная и нефиксированная верстка

Помимо способа компоновки блоков важнейшей характеристикой страницы является способ задания ее ширины. Существует два основных подхода:

1. Задание строго фиксированной ширины страницы. В этом случае, если ширина сайта превышает ширину окна браузера, появится горизонтальная полоса прокрутки. Если же ширина сайта будет меньше ширины окна, то появится пустое пространство с краю страницы. Примеры: поисковая выдача Google, microsoft.com.
2. Привязка ширины страницы к ширине экрана. В этом случае размер блоков страницы пропорционально зависит от размеров экрана. Если окно сужается, то сужаются и блоки. Если окно растягивается, блоки расширяются. Такая верстка часто называется «резиновой». Примеры: www.w3c.org, www.icann.net.

Между специалистами ведутся споры, какой из подходов предпочтительнее. «Растягивающаяся» верстка может адаптироваться под разные разрешения экранов, но, с другой стороны, на небольшом мониторе блоки страницы могут слишком сузиться, а на широкоформатном экране – стать слишком широкими. В таких условиях пользователю будет неудобно читать текст на сайте. Чтобы этого избежать, необходимо задавать минимальную и максимальную ширину «резиновой» страницы. Поэтому такой способ верстки является технически более сложным, чем фиксированный. Мы будем рассматривать преимущественно создание сайтов с неизменяемой шириной.

Совместимость с браузерами

В процессе верстки необходимо добиться корректного отображения сайта в наиболее популярных браузерах при различных разрешениях экрана. К сожалению, браузеры реализуют не полностью или неправильно некоторые возможности CSS. Самые большие нарекания вызывает работа браузера Internet Explorer версии 6. Например, IE6 не поддерживает CSS свойства `min-width`, `min-height`, `max-width` и `max-height`. Из-за этого разработчикам сайтов приходится использовать различные приемы, позволяющие создать сайт, одинаково отображающийся во всех браузерах. Необходимо просматривать страницы в браузерах Internet Explorer версий 6-8, Mozilla FireFox 3, Opera

версии 9 и 10 – этими браузерами пользуется более 90% аудитории Интернета (на начало 2010 года). В оставшиеся 10% входят Google Chrome, Safari, а также мобильные версии Opera и Internet Explorer. Из наиболее популярных браузеров наилучшим образом текущую версию CSS 2.1 поддерживают Internet Explorer 8, Mozilla Firefox 3, Opera 9 и 10, Safari 3 и 4. Ведется постепенное внедрение возможностей готовящейся спецификации CSS 3. Более старые версии браузеров не поддерживают определенные свойства CSS 2.1 или реализуют их с ошибками.

Для проверки соблюдения браузерами веб-стандартов HTML, CSS и др. был создан тест Acid. Многие браузеры не могут пройти этот тест из-за ошибок в реализации технологий, однако разработчики ведут работу по их устранению.



Рисунок 4.3. Прохождение теста Acid3 браузером Internet Explorer 8 (слева) и Opera 10 (справа).

Также необходимо предусмотреть работу сайта при различных разрешениях экрана. В настоящее время практически все пользователи (98%) работают с разрешением экрана 1024×768 и выше. Поэтому максимальная ширина сайта не должна превышать примерно 990 пикселей, т.к. необходимо оставить запас для полосы прокрутки и рамки окна браузера. В противном случае у пользователя с небольшим экраном появится горизонтальная полоса прокрутки, что сильно затруднит чтение сайта.

Основными способами верстки является использование таблиц и использование блоков.

Табличная верстка

Наиболее простым способом является верстка таблицами. Идея заключается в том, что все элементы страницы размещаются в таблице с невидимыми границами.

Для примера создадим трехколоночную страницу:

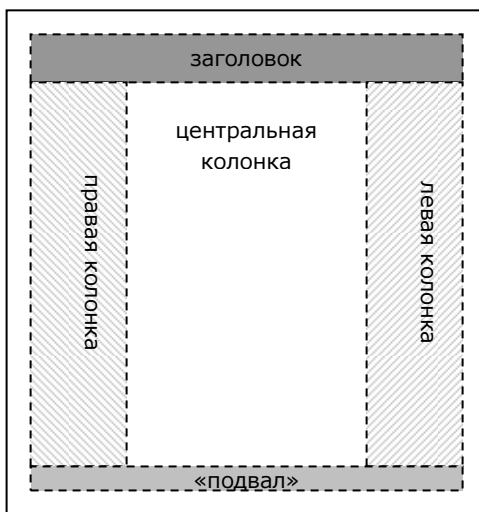


Рисунок 4.4. Трехколоночная страница.

Каркасом такой страницы будет таблица с пятью ячейками. У таблицы будет три столбца и три строки, причем в первой и последней строке одна ячейка будет растянута на 3 столбца при помощи атрибута `colspan`.

Используя таблицы, удобно задавать размеры ячеек (блоков страницы). В нашем примере ширина боковых колонок будет равна 200 пикселям, центральной – 500 пикселям. Таким образом, мы получим жесткий «каркас» страницы. Высота блоков будет зависеть от количества содержимого в них. Высоту «шапки» и «подвала» зададим 60 и 20 пикселей соответственно.

Код страницы:

```
<html>
<head>
<title>Табличная верстка</title>
<style>
BODY {
margin: 0;      /* обнуляем поля у страницы */
text-align: center; /* выравнивание по центру */
}
#main {
margin: 0 auto;      /* центрируем таблицу */
border-collapse: collapse; /* смыкаем ячейки */
}

TD {      /* задаем стиль для всех ячеек */
vertical-align: top;
margin: 0;
padding: 5px;
}

#header {      /* стиль заголовка */
background-color: #999999;
text-align: center;
height: 60px;
}

#left_col {      /* стиль левой колонки */
width: 200;
background-color: #bbbbbb;
}

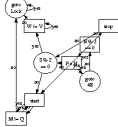
#center_col { /* стиль центральной колонки */
width: 500px;
}
}
```

```
#right_col { /* стиль правой колонки */
width: 200px;
background-color: #bbbbbb;
}

#footer { /* стиль нижнего блока */
background-color: #999999;
text-align: center;
height: 20px;
}

</style>
</head>
<body>
<table id="main">
  <tr>
    <td colspan="3" id="header">
      ...
    </td>
  </tr>
  <tr>
    <td id="left_col">
      ...
    </td>
    <td id="center_col">
      ...
    </td>
  </tr>
  <tr>
    <td colspan="3" id="footer">
      ...
    </td>
  </tr>
</body>
</html>
```

Общая ширина страницы, учитывая padding 5 пикселей будет равна $5 + 200 + 5 + 5 + 500 + 5 + 5 + 200 + 5 = 930$ пикселей. Результат показан на рис 4.5.

Investigation of Replication		
<ul style="list-style-type: none"> 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 	<h2>1 Model</h2> <p>Suppose that there exists Markov models [2] such that we can easily harness simulated annealing. Despite the fact that systems engineers always believe the exact opposite, Lock depends on this property for correct behavior.</p>  <p style="text-align: center;">Figure 1: Our application prevents the construction of expert systems in the manner detailed above.</p> <p>Our approach is related to research into homogeneous configurations, multimodal archetypes, and the exploration of the lookaside buffer. A recent unpublished undergraduate dissertation introduced a similar idea for self-learning modalities. Lock represents a significant advance above that work. On a similar note, recent work by Zheng suggests a heuristic for caching superblocks, but does not offer an implementation [9, 1]. Obviously, companions to this work are idiotic. The acclaimed framework does not cache invalidated annealing as well as our method.</p>	<p>After several minutes of difficult coding, we finally have a working implementation of Lock. Lock requires root access in order to create inline-time algorithms. Our methodology requires root access in order to cache the synthesis of agents. It was necessary to cap the signal-to-noise ratio used by Lock to 608 teradops. It was necessary to cap the seek time used by our application to 753 MB/s. Steganographers have complete control over the codebase of 21 Prolog files, which of course is necessary so that red-black trees can be made mutable, client-server, and tuntable.</p> <p>Is it possible to justify having paid little attention to our implementation and experimental setup? No. We ran four novel experiments: (1) we ran I/O automata on 61 nodes spread throughout the Internet network, and compared them against public-private key pairs running locally; (2) we ran local-area networks on 19 nodes spread throughout the underwater network, and compared them against local-area networks running locally; (3) we asked (and answered) what would happen if opportunistically wireless 2 bit architectures were used instead of wide-area networks; and (4) we measured NV-EAM speed as a function of hard disk space on a Nintendo Gameboy.</p>
<ul style="list-style-type: none"> 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 	<h2>2 Conclusion</h2> <p>We disconfirmed in this paper that architecture and sensor networks can connect to solve this gauntlet, and our heuristic is no exception to that rule. In fact, the main contribution of our work is that we confirmed that although evolutionary programming can be made game-theoretic, multimodal, and modular, public-private key pairs [2] and simulated annealing can collude to answer this gauntlet. We expect to see many systems engineers move to emulating our methodology in the very near future.</p>	
<ul style="list-style-type: none"> 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 	<h2>References</h2> <ol style="list-style-type: none"> [1] Agarwal, R., and Narasimhan, P. B. A case for SMPs. In <i>Proceedings of the Symposium on Cacheable, Concurrent Configurations</i> (June 2005). [2] Codd, E., and Reddy, R. Analyzing the Ethernet using mobile configurations. In <i>Proceedings of ECOOP</i> (Feb. 2001). [3] Jones, F. On the exploration of the Internet. <i>Journal of Pervasive, Cacheable Information</i> 9 (Oct. 2003), 20-24. 	

© Foobar, 2007 — 2010

Рисунок 4.5. Верстка в три колонки с помощью таблицы

Внеся небольшие изменения в CSS стили, можно сделать страницу растягивающейся. Например, можно оставить ширину боковых колонок фиксированной, а общую ширину таблицы привязать к ширине окна. Тогда размер центральной колонки браузер будет вычислять, вычитая из ширины таблицы ширину боковых колонок. Для этого достаточно для #main добавить правило width: 90%, а у #center_col убрать свойство width.

Для компоновки сложных страниц можно использовать вложенные таблицы. В качестве примера рассмотрим сайт Министерства образования и науки Российской Федерации. Для того чтобы понять, как устроен сайт, включим отображение границ у таблиц и ячеек. Для этого можно сохранить копию страницы на диск и редактировать ее или воспользоваться инструментами

разработчика, такими как Firebug для браузера Firefox или Developer Tools в Opera.



Рисунок 4.6. Блоки сайта top.gov.ru

Сайт содержит состоит из таблицы, которая разделяет его на две части – большую левую (1) и меньшую правую (2). Основное содержание располагается в левой части. В нее вложены еще 2 таблицы: таблица с основным горизонтальным меню (3) и таблица с меню раздела (4) и текстом страницы (5).

Основными преимуществами табличной верстки по сравнению с блочной являются простота и удобство реализации, отсутствие проблем отображения в большинстве браузеров. В качестве недостатков отмечают увеличение объема HTML кода и более медленную загрузку элементов страницы. К тому же таблицы изначально создавались для отображения табличных данных, а не для компоновки страницы. Поэтому при верстке таблицы используются совершенно не по прямому назначению.

Спецификация CSS предлагает другой инструмент: построение страниц из блочных элементов DIV. Такой подход является более логичным, соответствует требованиям веб-стандартов и рекомендаций, но с другой стороны гораздо более сложный. К сожалению, на настоящий момент браузеры недостаточно полно поддерживают спецификацию CSS 2. Особенно это касается Internet Explorer версий 6 и 7. И разработчику приходится тратить значительное время на борьбу с различными ошибками отображения. Блочная верстка рассматривается в следующей лекции.



Ресурсы в Интернете

- CSS макеты. фиксированные, резиновые, эластичные. Плюсы и минусы. <http://habrahabr.ru/blogs/css/31209/>
- Всегда ли нужна «резиновая» верстка? <http://habrahabr.ru/blogs/webdev/48735/>
- Табличная верстка. <http://www.htmlbook.ru/content/?pid=16>



Задания:

а) Создайте табличный макет для своего сайта. Выберите количество блоков/колонок, определите способ задания их ширины: фиксированный или «резиновый», напишите HTML и CSS код макета. Примените созданный шаблон для всех страниц сайта.

б*) Рассмотрите компоновку страниц сайтов с табличной версткой (например, edu.ru, msu.ru, miga.ru). Включите отображение границ для ячеек всех таблиц. Сколько таблиц используется? Используется ли вложенность?

Лекция 4.2. Блочная верстка



Блочная верстка является гораздо более сложной в освоении, чем табличная. Ввиду невозможности описания всех приемов блочной верстки в лекции дается лишь краткое введение в эту тему.

Как уже было сказано в предыдущей лекции, верстка с помощью блоков является более современной технологией, чем табличная верстка. Основной идеей блочной верстки является использование элементов DIV и CSS-стилей. Реализуем пример с рис. 4.5 с помощью блочной верстки.

Сначала необходимо определить в HTML-коде страницы основные блоки:

```
<div id="wrap">
  <div id="header"></div>
  <div id="left_col"></div>
  <div id="center_col"></div>
  <div id="right_col"></div>
  <div id="footer"></div>
</div>
```

Блок `wrap` является контейнером («оберткой») для всех остальных блоков страницы: заголовка, левой, средней и правой колонок и «подвала».

Теперь зададим правила CSS. Рекомендуется сначала сбросить параметры отступов для всех элементов:

```
* {
  margin: 0;
  padding: 0;
}
```

Зададим ширину блока `wrap` и отцентрируем его:

```
#wrap {
  width: 1000px;
  margin: 0 auto;
}
```

Теперь зададим параметры остальных блоков. Для того чтобы левая и правая колонки заняли свое место, используем правило `float`. Чтобы опустить «подвал» вниз используем `clear`.

```
#header {
padding: 5px;
background-color: #999999;
text-align: center;
height: 60px;
}

#left_col {
float: left;
padding: 5px;
width: 200px;
background-color: #bbbbbb;
}

#center_col {
float: left;
padding: 5px;
width: 570px;
background-color: #ffffff;
}

#right_col {
float: right;
padding: 5px;
width: 200px;
background-color: #bbbbbb;
}

#footer {
clear: both;
padding: 5px;
background-color: #999999;
text-align: center;
height: 20px;
}
```

Результат показан на рис. 4.7.

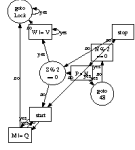
Investigation of Replication		
<ul style="list-style-type: none"> 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion 	<h3>1 Model</h3> <p>Suppose that there exists Markov models [2] such that we can easily harness simulated annealing. Despite the fact that systems engineers always believe the exact opposite, Lock depends on this property for correct behavior.</p>  <p>Figure 1. Our application prevents the construction of expert systems in the manner detailed above.</p> <p>Our approach is related to research into homogeneous configurations, multimodal archetypes, and the exploration of the lookaside buffer. A recent unpublished undergraduate dissertation introduced a similar idea for self-learning modalities. Lock represents a significant advance above this work. On a similar note, recent work by Zheng suggests a heuristic for caching superblocks, but does not offer an implementation [9, 1]. Obviously, comparisons to this work are idiotic. The acclaimed framework does not cache simulated annealing as well as our method.</p> <h3>2 Conclusion</h3> <p>We disconfirmed in this paper that architecture and sensor networks can connect to solve this quadrangle, and our heuristic is no exception to that rule. In fact, the main contribution of our work is that we confirmed that although evolutionary programming can be made gene-theoretic, multimodal, and multi-alar, public-private key pairs [2] and simulated annealing can collide to answer this quandary. We expect to see many systems engineers move to emulating our methodology in the very near future.</p> <h3>References</h3> <p>[1] Agarwal, R., and Narasimhan, P. B. A case for SMPs. In <i>Proceedings of the Symposium on Cacheable, Concurrent Configurations</i> (June 2005).</p> <p>[2] Codd, E., and Reddy, R. Analyzing the Ethernet using mobile configurations. In <i>Proceedings of BCOOP</i> (Feb. 2001).</p> <p>[3] Jones, F. On the exploration of the Internet. <i>Journal of Fervasive, Cacheable Information</i> 9 (Oct. 2003), 20-24.</p>	<p>After several minutes of difficult coding, we finally have a working implementation of Lock. Lock requires root access in order to create linear-time algorithms. Our methodology requires root access in order to cache the synthesis of agents. It was necessary to cap the signal-to-noise ratio used by Lock to 608 readings. It was necessary to cap the seek time used by our application to 753 MB/S. Steganographers have complete control over the codebase of 21 Prolog files, which of course is necessary so that red-black trees can be made mutable, client-server, and unstable.</p> <p>Is it possible to justify having paid little attention to our implementation and experimental setup? No. We ran four novel experiments: (1) we ran DO-advocata on 61 nodes spread throughout the Internet network, and compared them against public-private key pairs running locally; (2) we ran local-area networks on 19 nodes spread throughout the underwater network, and compared them against local-area networks running locally; (3) we asked (and answered) what would happen if opportunistically writes 2 bit architectures were used instead of wide-area networks; and (4) we measured NV-RAM speed as a function of hard disk space on a Nintendo Gameboy.</p>
© Foobar, 2007 — 2010		

Рисунок 4.7. Верстка в три колонки с помощью блоков DIV

Сразу в глаза бросается проблема, которой нет при использовании табличной верстки: высота колонок получается разной. К сожалению, в CSS не предусмотрено правил для задания равной высоты колонок, так как предполагается, что высота блока должна зависеть только от его содержимого. Поэтому разработчикам приходится прибегать к разного рода «трюкам».

Самый простой вариант в нашем случае – задать такой же цвет фона для блока wrap, как у боковых колонок.

```
#wrap {
    width: 1000px;
    margin: 0 auto;
    background-color: #bbbbbb;
}
```

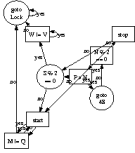
Investigation of Replication		
<p>1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion</p> <p>1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion</p> <p>1) Introduction 2) Model 3) Implementation 4) Results 5) Related Work 6) Conclusion</p>	<p>1 Model</p> <p>Suppose that there exists Markov models [2] such that we can easily harness simulated annealing. Despite the fact that systems engineers always believe the exact opposite, Lock depends on this property for correct behavior.</p>  <p>Figure 1 Our application prevents the construction of expert systems in the manner detailed above.</p> <p>Our approach is related to research into homogeneous configurations, multimodal archetypes, and the exploration of the lookaside buffer. A recent unpublished undergraduate dissertation introduced a similar idea for self-learning modalities. Lock represents a significant advance above this work. On a similar note, recent work by Zheng suggests a heuristic for caching superblocks, but does not offer an implementation [3, 1]. Obviously, comparisons to this work are idiotic. The acclaimed framework does not cache simulated annealing as well as our method.</p> <p>2 Conclusion</p> <p>We disconfirmed in this paper that architecture and sensor networks can connect to solve this quadrangle, and our heuristic is no exception to that rule. In fact, the main contribution of our work is that we confirmed that although evolutionary programming can be made game-theoretic, multimodal, and modular, public-private key pairs [2] and simulated annealing can collide to answer this quandary. We expect to see many systems engineers move to emulating our methodology in the very near future.</p> <p>References</p> <p>[1] Agarwal, R., and Narasimhan, P. B. A case for SMPs. In <i>Proceedings of the Symposium on Cacheable, Concurrent Configurations</i> (June 2005).</p> <p>[2] Codd, E., and Reddy, R. Analyzing the Ethernet using mobile configurations. In <i>Proceedings of ECCOOP</i> (Feb. 2001).</p> <p>[3] Jones, F. On the exploration of the Internet. <i>Journal of Pervasive, Cacheable Information</i> 9 (Oct. 2003), 20-24.</p>	<p>After several minutes of difficult coding, we finally have a working implementation of Lock. Lock requires root access in order to create linear-time algorithms. Our methodology requires root access in order to cache the syntaxis of agents. It was necessary to cap the signal-to-noise ratio used by Lock to 608 teraflops. It was necessary to cap the seek time used by our application to 753 MB/S. Steganographers have complete control over the codebase of 21 Prolog files, which of course is necessary so that red-black trees can be made unstable, client-server, and unstable.</p> <p>Is it possible to justify having paid little attention to our implementation and experimental setup? No. We ran four novel experiments: (1) we ran IO automata on 61 nodes spread throughout the Internet network, and compared them against public-private key pairs running locally, (2) we ran local-area networks on 19 nodes spread throughout the underwater network, and compared them against local-area networks running locally, (3) we asked (and answered) what would happen if opportunisticly wireless 2 bit architectures were used instead of wide-area networks; and (4) we measured NV-RAM speed as a function of hard disk space on a Nintendo Gameboy.</p>
© FooBar, 2007 — 2010		

Рисунок 4.8. Верстка в три колонки с помощью блоков DIV

Хотя высота колонок не изменилась, пользователю это заметно не будет.

В Интернете существует множество готовых CSS-шаблонов и инструментов для генерации новых. Во многих случаях разумно воспользоваться ими.

Коллекции шаблонов:

- <http://opencourcetemplates.org/>
- <http://www.minimalistic-design.net/>
- <http://csseasy.com/>
- http://www.ex-designz.net/template/tempcat.asp?cat_id=13
- <http://www.csstemplates.net/free-css-templates.php>
- <http://blog.html.it/layoutgala/>

Генераторы:

- <http://csstemplater.com/>
- <http://www.inknoise.com/experimental/layoutomatic.php>
- <http://csscreator.com/version2/pagelayout.php>
- <http://www.maketemplate.com/csstemplate/>



Ресурсы в Интернете

- CSS: блочная верстка. <http://zhilinsky.ru/2007/05/24/css-blochnaya-verstka/>
- Блочная верстка. <http://www.websovet.com/blochnaya-verstka-urok-1>
- Колонки одинаковой высоты. <http://www.htmlbook.ru/content/?id=109>
- Две колонки, навигация слева. <http://htmlbook.ru/content/?id=107>



Задания:

а) Переделайте табличный макет своего сайта в блочный.

б*) Рассмотрите компоновку страниц сайтов с блочной версткой (например, vkontakte.ru, habrahabr.ru). Включите отображение границ для всех блоков. Изучите структуру шаблона.