МИНИСТЕРСТВО НАУКИ И ВЫШЕГО ОБРАЗОВАНИЯ РФ
ФГБОУ ВПО «ДАГЕСТАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

КУРС «ДИЗАЙН»

УЧЕБНО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ

для выполнения лабораторных работ по дисциплине «Основы Web-дизайна» для студентов направления подготовки бакалавров 09.03.03. –«Прикладная информатика», профиль «Прикладная информатика в дизайне»

Махачкала – 2021

УДК:

Учебно-методические указания для выполнения лабораторных работ по дисциплине «Основы Web-дизайн» для студентов направления подготовки бакалавров 09.03.03. –«Прикладная информатика» профиль «Прикладная информатика в дизайне».- Махачкала, ДГТУ, 2021. – 56 с.

В учебно-методических указаниях рассматриваются современные веб-технологии клиентской стороны: язык разметки HTML в сочетании с каскадными листами стилей CSS2.

Методические рекомендации позволяют сложить целостное представление о технологической цепочке создания веб-сайтов и формируют понимание актуальных тенденций развития веб-технологий. Содержат большое количество практических примеров и скриншотов. Среди примеров имеется ряд завершенных разработок, обладающих самостоятельной практической ценностью. Приведены индивидуальные задания, имеющие различные уровни сложности. В курсе также рассмотрены вопросы графического представления сайтов и аспекты, связанные с их художественным оформлением. Приведены также сведения об особенностях коммерческих сайтов, используемых для электронного бизнеса. Рассмотрены эргономические и эстетические аспекты создания сайтов и основные принципы Web-дизайна

Составители:

1. **Парамазова А.Ш. – зав. курсом «Дизайн»**
2. **Рамазанов Г.М.. – ст. преподаватель**

Рецензенты:

1. Курбанов М.К. – директор авторизированного учебного центра «Академия компьютерной графики»

2. Исабекова Т.И.– зав. кафедрой ПМиИ ДГТУ, к.ф-м.н., доцент

Печатается по решению Совета Дагестанского государственного технического университета от «____» _____ 2021 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Любую веб-страницу можно разложить на три составляющие. Это ее непосредственное содержание, структура и представление. Один и тот же документ может иметь множество представлений: например, его внешний вид может различаться в зависимости от того, где документ отображается: на экране настольного компьютера или ноутбука, на экране карманного компьютера или же на листе бумаги при печати. Помимо визуальных представлений, определяющих внешний вид, у документа могут быть и другие виды представлений, например, аудиовизуальные, то есть звуковые.

Тенденции развития языка разметки нацелены на торжество принципа разделения содержания и представления на уровне конечного кода веб-страниц, передаваемого веб-сервером клиенту. При соблюдении этого принципа на HTML возлагается лишь задача структурирования контент, тогда как управление его представлениями полностью ложится на плечи CSS.

Учебно-методические указания к выполнению лабораторных работ по дисциплине «Основы Web-дизайна» посвящен рассмотрению основных технологий создания Web-сайтов – языка разметки документов HTML, выполняющего структурную разметку Web-страниц, каскадных стилевых спецификаций (CSS), формализующих процесс форматирования страниц, объектных моделей браузера (BOM) и документа (DOM), позволяющих сценариям обращаться к элементам страниц и объектам браузера, языка создания сценариев JavaScript, обеспечивающего интерактивность создаваемого сайта.

Теоретические сведения к лабораторным занятиям

1. Технологии и стандарты World Wide Web

Интерактивная мультимедийная гипертекстовая среда **World Wide Web (WWW**) появилась в современном виде благодаря наличию нескольких факторов. Первый из них−платформенно - независимый способ распространения информации.

Как известно, компьютеры совместно с операционными системами (DOS, семейство Windows, Windows NT, Mac OS, семейство UNIX, Linux и др.) образуют "*платформы*", на которых функционируют программы, реализующие разнообразные "*пользовательские среды*" (графические, текстовые, звуковые).

Для успешного взаимодействия в сети необходимы средства, не зависящие от конкретной компьютерной платформы. Такими средствами являются единое адресное пространство и сетевые протоколы.

Другая важная задача обеспечения эффективного взаимодействия компьютеров в сети связана с проблемой доступа в сеть, и, соответственно, с доступностью информации. В настоящее время доступ в сеть обеспечивается далеко не только с помощью компьютеров. Мобильные телефоны, карманные компьютеры, Web TV обладают такой возможностью, но отличаются от компьютеров размером экрана и устройствами ввода. Доступ в Internet различается также и в зависимости от способа представления информации, который отличен в текстовых, графических и речевых браузерах. Решение задачи связано с воплощением идеи разделения *структуры* и содержания документов от их *представления*. Представление может отличаться в разных устройствах доступа, тогда как структура и содержание остаются неизменными.

Следует также отметить совершенствование технологии "клиент-сервер", связанное с появлением графических многофункциональных браузеров, клиентских и серверных расширений, обеспечивающих эффективное интерактивное взаимодействие, а также совершенствование языков написания сценариев и серверных программ.

1.1. WWW

WWW образует *технологию обмена* гипертекстовыми документами. В ее основу положено использование идеи *гипертекста*, включающего в себя разнотипные данные, объединенные механизмом *гиперссылок*, позволяющим интегрировать документы, размещенные на разных компьютерах в разных сетях.

Документы WWW называют также Web-страницами, Web-документами, HTML-файлами. Эти документы имеют стандартное расширение html или htm.

Основная особенность гипертекстовых документов – наличие гиперссылок, которые позволяют преодолеть ограничения на размер документа (hyper - сверх). Таким образом, WWW можно представить в виде гигантской библиотеки гипертекстовых, интерактивных, мультимедийных документов, связанных гиперссылками.

В общем виде гиперссылка указывает на документ, находящийся на сервере и имеет следующие части:

http://www.server_name/directories/file_name

В первой части указан протокол, который использует Web-сервис, затем имя сервера, директории, ведущие к файлу и имя целевого файла.

WWW использует для обмена данными протокол HTTP (HyperText Transfer Protocol, протокол обмена гипертекстом). Это высокоуровневый протокол (прикладного уровня), который работает «поверх» низкоуровнего протокола TCP/IP (Transfer Control Protocol/Internet Protocol, протокол управления обменом, протокол Internet).

Информационной единицей WWW является *сайт* (site – площадка) – площадка для размещения Web-страниц. Сайт – это совокупность Web-страниц, связанных одной темой, единым оформлением и имеющая имя.

Имя имеет следующий вид:

http://www.server_name/directories/index.html

После указания протокола следует доменное имя сервера, а затем, если необходимо, указываются папки и имя домашней страницы сайта. По умолчанию, домашняя страница имеет имя index.html или default.html.

Процесс размещения сайта на сервере является *публикацией* сайта. Поддержка работы сайта, серверных программ, а также баз данных, задействованных в работе сайта, называется *администрированием* сайта.

Область знаний, предметом которой является разработка (development) и представление (design) Web-страниц называется *Web-дизайном*.

1.2. WWW Консорциум

В 1994 году был создана специальная, ныне международная и некоммерческая организация Консорциум W3 (W3 Consortium, сокращенно W3C), занимающаяся разработкой стандартов для протоколов и языков, связанных с Web. В настоящее время консорциум объединяет свыше 150 организаций, в том числе Microsoft**,** и множество других. Консорциум имеет официальный сайт в Internet – http://www.w3c.org/, где публикуется самая последняя информация, связанная со стандартами Web.

Можно выделить три основные цели этой организации:

- обеспечение универсального доступа к сети, так, чтобы сетью мог пользоваться каждый;
- разработка программных средств, позволяющих пользователям взаимодействовать в сети;
- управление развитием сети с учетом юридических, социальных или коммерческих аспектов.

Каждая новая версия спецификации любой сетевой технологии, прежде чем стать стандартом, подлежит тщательному рассмотрению в W3C.

1.3. SGML

В конце шестидесятых годов в фирме IBM был разработан язык разметки технической документации, который приобрел в 1986 году статус международного стандарта – SGML (StandardGeneralizedMarkupLanguage). Этот обобщенный язык был предназначен для построения систем логической, *структурной* разметки любых разновидностей текста независимо от компьютерных платформ.

Структурная разметка означает, что управляющие коды, производящие такую разметку, не несут никакой информации о форматировании документа, а лишь указывают границы и соподчинение его основных частей, т.е. задают его *структуру*. Форматирование при этом выносится в так называемые стилевые спецификации – отдельный и допускающий независимые изменения информационный слой. Благодаря этому размеченный текст может интерпретировать любая программа, работающая с любым устройством вывода. Этот аспект тесно связан с обеспечением *доступности* информации в Internet не только для любых платформ, но и сред, которые не ограничиваются распространенной *графической* средой, а включают в себя и *текстовые* и *звуковые* среды.

1.4. Языки разметки HTML, XML, XHTML

Разметка Web-документа осуществляется с помощью языков разметки. Разметка включает в себя описание структуры, в которой размещен content (содержание)

документа. Внешний вид документа описывается с помощью самостоятельной технологии – каскадных таблиц стилей (Cascading Style Sheets, CSS).

HTML (Hyper Text MarkUp Language, язык разметки гипертекста) закончил свое развитие и ныне существует в своей окончательной версии 4.01. С ее стандартами можно познакомиться на сайте W3C. Дальнейшее развитие языков разметки возможно на основе XML (eXtansible MarkUp Language, расширяемый язык разметки) и его производных.

HTML имеет один вид управляющих конструкций – теги (или дескрипторы). Они используются для описания структуры и для внедрения объектов на странице, таких как графические изображения, аудио-клипы или видео-ролики.

Элемент HTML состоит из следующих частей (на примере элемента p – параграфа):

content

открывающего тега с именем p, его атрибутов align=”center” id=”para1”, содержимого (content) и закрывающего тега.

W3C стандартизировал набор логических тегов, которые должны заменить собой теги физического форматирования как не отвечающие требованиям разделения структуры и представления, а также универсальности. К примеру, тег физического форматирования <b> (bold), обеспечивающий полужирное начертание, не имеет смысла в речевом браузере, тогда как логический тег позволяет выделить контент в зависимости от типа среды, в которой он используется.

Традиционный HTML определяется шаблоном DTD (Document Type Definition, описание типа документа), который определяет набор тегов, их атрибуты, вложенность тегов и другие правила использования языка.

Набор тегов HTML фиксирован и ограничен, поэтому для HTML используется одно DTD, ссылка на которое выглядит следующим образом:

<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//En”

Сделать развитие языка естественным процессом, позволив браузеру обрабатывать теги, изобретенные разработчиком, можно используя XML (eXtansible MarkUp Language, расширяемый язык разметки) и его производные. Для браузеров мобильных телефонов аудио подача информации, которая может осуществляться, например, синтезатором речи, выглядит вполне естественно. Синтезатор речи будет требовать какой-то определенный, особенный набор команд. Для мощных графических станций акценты делаются на трехмерную графику, для которой может быть введена своя система команд.

В настоящее время широко применяются, созданные на основе XML язык WML (Wireless MarkUp Language, язык беспроводной связи) для мобильной связи или MathML (язык разметки математических текстов) для передачи математических документов.

Языки, созданные на основе XML должны иметь соответствующие описания, которые помещаются в DTD или XML-схему. В отличие от HTML каждый язык должен обладать уникальным DTD. Оформление документов возможно только с использованием CSS или специального языка XSL (eXtansible Stylesheets Language, расширяемый язык стилевых спецификаций). Еще одной особенностью являются более строгие, чем в HTML синтаксические правила, что обеспечивает более простую машинную обработку.

Формальные изменения касаются более строгих правил разметки, которые устанавливают использование только парных тегов, нижнего регистра для записи тегов и атрибутов, заключение атрибутов в кавычки, а также выполнение правила вложенности тегов. Оформление документа возможно только с применением CSS.

В документе XHTML обязательны следующие теги:

<?xml version 1.0?> - объявляет документ документом XML

<!doctype…..> - ссылка на соответствующее DTD

<html xmlns=”…”> - ссылка на пространство имен

<head>….

1.5. Каскадные таблицы стилей (Cascading Style Sheets, CSS)

Язык HTML был создан для разметки научных текстов и не содержал первоначально средств оформления документов. С тех пор как Internet стал достоянием широкой общественности появилась потребность в придании страницам гипертекста более привлекательного вида. Эта потребность побудила фирмы – производители браузеров создавать элементы HTML, назначением которых было форматирование документа. В силу коммерческих интересов были созданы многочисленные теги, которые корректно отображались только в браузерах производителей. В результате одна и та же Web-страница выглядела по разному в разных браузерах.

Для того, чтобы вернуть HTML его истинное назначение и ликвидировать произвол в отображении документов W3C разработал технологию, независимую от HTML и предназначенную для *представления* документа. Эта технология получила название «Каскадные таблицы стилей (Cascading Style Sheets, CSS)». CSS первого уровня появились в 1995 году. В настоящее время созданы стандарты CSS2 и CSS3. Последние разработки браузеров поддерживают CSS2 и частично CSS3.

Важной частью стандарта является ввод понятия типа среды – текстовой, графической, звуковой, и конкретизация средств оформления страницы, соответствующих этим типам. Так, в звуковом типе среды определены свойства, позволяющие регулировать *громкость*, *темп* произнесения текста и *тембр* голоса. Таким образом, одна и та же структура и содержание документа по-разному представляется в зависимости от оформительских средств данного типа среды.

Из графических нововведений можно отметить возможность использовать *подключаемые шрифты*, относительное и абсолютное позиционирование объектов, а также использование пространственных эффектов.

Технология CSS формально независима от HTML, имеет совершенно иной синтаксис и позволяет задавать параметры графического (также как и текстового, и звукового) представления дескрипторов HTML.

Достоинствами CSS можно назвать:

обеспечение возможности представления документа способом, независимым от описания структуры;

гибкость, позволяющая при внесении изменений в файле описания стилевых свойств, связанного со страницами сайта, синхронно внести изменения во все связанные страницы;

значительное расширение функциональных возможностей оформления контента.

Подключение стилей к документу может быть произведено тремя способами: встраиванием – описание стиля включается как атрибут к тегу HTML и действует только на этот тег; внедрением – описание стилей вынесено в заголовок документа и относится ко всей странице, и, наконец, связыванием, при котором стили определяются в отдельном файле, и этот файл связывается со страницами сайта, тем самым воздействуя на весь сайт.

При этом установлен приоритет стилей в порядке убывания: встраивание - внедрение – связывание. Приоритетностью стилей объясняется название стилей «каскадные».

1.6. JavaScript

Появление языков сценариев, интерпретируемых браузером, связано с восстребован ностью интеллекта не только на сервере, но и на клиентской стороне.

Скриптовой язык JavaScript поддерживался уже первым браузером, выпущенным фирмой Netscape. Он преобразовался из языка, имевшего первоначальное название

LiveScript, и предназначен для управления содержимым страницы и собственно браузером.

Такие задачи как проверка данных формы перед ее отправкой на сервер, помещение страниц в окна с произвольным расположением и размером, создание меню и простой анимации решаются с помощью сценарного языка на клиентской стороне. В Web-страницах используются языки JavaScript, получивший статус стандартного, и VBScript, который является разработкой фирмы Microsoft и поддерживается браузером Internet Explorer.

Язык JavaScript является интерпретируемым (отсюда название «язык сценариев» в отличие от языков программирования, которые являются компилируемыми) языком, ориентированным на обработку событий. Он относится к группе объектно-ориентированных языков (ООЯ) и строится на работе с объектами.

JavaScript в настоящее время широко используется и на серверной стороне. В частности, в составе активных серверных страниц (ASP, Active Server Pages) содержится HTML-код и фрагменты, написанные на скриптовых языках. После выполнения кода таких страниц клиенту отсылается остаток в виде HTML-кода, который и воспроизводится браузером.

Язык JavaScript в сочетании с объектной моделью документа (DocumentObjectModel, DOM) и каскадными таблицами стилей позволяет управлять элементами страницы и их свойствами после загрузки страницы в браузер. Эту комбинацию Web-технологий, основанную на различных стандартах (CSS, DOM, JavaScript и языков разметки) принято называть DHTML (Dynamic HTML, динамический HTML). DHTML позволяет существенно увеличить интерактивность Web-страницы. Созданная на основе DHTML страница может изменяться, не обращаясь к серверу за дополнительными данными.

Стандартизацией языка JavaScript занимается ECMA (European Computer Manufacturers Association). Его "официальная" версия называется ECMAScript**.**

1.7. Объектная модель документа

W3C разработал стандарт "объектной модели документа" (Document Object Model, DOM).

С помощью объектной модели документа (Document Object Model, DOM) можно динамически менять Web-страницу, используя язык написания сценариев. DOM ставит в соответствие каждому элементу определенный идентификатор - атрибут ID. Идентификация позволяет элементу получить свой собственный уникальный адрес, а также преобразовать элемент в объект. Определяя элемент как объект, можно менять любое его свойство, заданное с помощью CSS или атрибута, а также сам элемент или даже его контент. Последняя версия стандарта DOM предусматривает мощный событийный интерфейс. Основные браузеры поддерживают последний стандарт DOM.

2. Технологии создания Web-страниц. Язык разметки гипертекста HTML

2.1. Синтаксис

Синтаксис языка **HTML** (**Hyper Text Markup Language**) достаточно прост. Кроме обычного текста **HTML**-файл содержит только один тип управляющих конструкций, так называемые *дескрипторы* (или *теги*). Дескрипторы разделяют исходный неформатированный текст на *элементы* и создают новые элементы, например, графические вставки или **Java**–аплеты.

Элементы бывают двух видов – *парные*, охватывающие какой-то фрагмент текста или (и) другие элементы и *непарные*. Например, парный элемент <**h1**> используется для создания заголовка первого уровня: **<h1>текст заголовка</h1>**, непарный элемент

<**img**>используется для размещения на странице графического изображения: **<imgsrc="URL">**.

Парные элементы должны вкладываться друг в друга без пересечений, т.е. если в области действия элемента А открылся элемент В, то элемент В должен закрыться до того, как закроется элемент А.

Многие элементы имеют *атрибуты*, которые уточняют и изменяют значение элемента. В большинстве случаев атрибуты не являются обязательными и интерпретатор **HTML** должен использовать значения по умолчанию. Атрибуты включаются в элементы в виде пары *атрибут="значение"* и отделяются друг от друга пробелами. Например, элемент **img** может иметь несколько атрибутов:

<imgsrc="URL" alt="текст" width="значение" height="значение">

Согласно стандарту, кавычки вокруг значения атрибута обязательны в тех случаях, когда значение содержит какие-либо символы, кроме букв, цифр, точки или дефиса. Регистр букв в идентификаторах элементов и атрибутов не имеет значения, но с точки зрения совместимости с **XML** лучше всегда пользоваться кавычками и использовать нижний регистр.

Разметка текста происходит с помощью элементов, которые удобно расположить по группам в зависимости от их функционального назначения:

1. элементы *структуры* документа (заголовки шести уровней **<h1>…<h6>**, выделение блока, или слоя** **, встроенный блок, или фрагмент ****, абзац **<p>**, цитата **<blockquote>**, перевод строки **
, **<nobr>, **<wbr>**, списки нумерованные **
…
**, списки маркированные **
…
, списки-определения **<dl>, <dt>, <dd>, вывод отформатированного текста **<pre>**, разделитель **<hr>**, информация об авторе **<address>**);

2. элементы *физического* форматирования (курсив **…**, полужирное начертание **…**, моноширинный шрифт **<tt>…</tt>**, уменьшенный кегль **<small>…</small>**, увеличенный кегль **<big>…</big>**, зачеркивание **<strike>…</strike>** или **<s>…</s>**, подчеркивание **<u>…</u>**) и *логического* форматирования (курсив **…**, полужирноеначертание**…,** код программы моноширинным шрифтом**<code>…</code>,** выделяет переменные в коде программы**<var>…</var>,** выделяет текст, введенный с клавиатуры **<kbd>…</kbd>,** термины – определители **<dfn>…</dfn>,** цитаты **<cite>…</cite>,** аббревиатура**<abbr>…</abbr>**)

3. элементы *таблиц* (таблица **<table>…</table>,** заголовок таблицы**<caption>…</caption>,** заголовки столбцов вверху таблицы **<thead>…</thead>**, заголовки столбцов внизу таблицы **><tfoot>…</tfoot>**,тело таблицы **<tbody>…</tbody**, строка **<tr>…</tr>**, ячейка **<td>…</td>**, полужирное начертание в ячейке **<th>…</th>**, организация колонок <**col**> и <**colgroup**>…</**colgroup**>);

4. элемент *ссылок* (<**a**>),

5. элементы*форм***(<form>…</form>, <input>,<select><option>…</option><select>)**

6. элементы*фреймов***(<frameset>…</frameset>, <frame>, <noframe>**),

7. элементы*графических*объектов (**<img>** (см.п.6.2)), *мультимедиа* (**<object>,**<**embed**>) и*сценариев* (<**javascript>…</javascript>**).

2.2. Инструменты HTML

2.2.1. Текстовые редакторы

Информация **HTML** хранится в обычном текстовом формате, поэтому для кодирования **HTML**-документов можно пользоваться обычными текстовыми редакторами. В среде **Windows** простейшим редактором является **Notepad** (блокнот).

Для **UNIX** используются редакторы **vi** или **Pico**. Этот способ кодирования требует досконального знания **HTML**, однако, он лишен таких недостатков, как избыточность кода или его модификация, которые зачастую имеют место при использовании приложений **WYSIWYG**. Только при кодировании в текстовой среде разработчик имеет полный контроль над кодом страницы.

2.2.2. Специализированные редакторы

Специализированные текстовые редакторы занимают промежуточное место между простым текстовым редактором и редактором **WYSIWYG.**

Обычно в них используется простой графический интерфейс и большое количество мощных инструментов. Эти средства желательно применять, обладая значительными знаниями **HTML**, когда требуется работать более эффективно. Процесс создания кода ускоряется за счет наличия таких инструментов, как, например, проверка орфографии.

В специализированных текстовых редакторах предлагаются шаблоны, инструментальные панели для вставки элементов, а также мастера изображений, которые автоматически вставляют в код их размеры.

Из данной группы редакторов можно отметить **HotDogPro**, **HomeSite** для **Windows**, **BBEdit** и **WebWeaver**–для **Macintosh.**

2.2.3. WYSIWYG редакторы

Приложения данной группы также обладают многофункциональным графическим интерфейсом, позволяющим размещать необходимую графику и текст так, как эти элементы должны выглядеть на **Web**-странице. За последующую генерацию кода отвечает программное обеспечение. Некоторые приложения **WYSIWYG** имеют мощные дополнительные возможности, не связанные с созданием страниц. К таким возможностям относятся *расширения*, позволяющие управлять проектами или специальные приложения для поиска и поддержки форм, как в программе **FrontPage** фирмы **Microsoft.**

Недостатками данных редакторов является их сложность, а также генерация избыточного кода. Кроме того, приложения **WYSIWYG** зачастую не успевают отслеживать те изменения, которые происходят в среде **WWW**. Появление новых форматов файлов, подключаемых модулей и функций браузеров требует своевременного обновления приложений **WYSIWYG.**

Однако эти программы в ряде случаев могут быть очень эффективными, например, создание фреймов или сложных таблиц значительно упрощается при использовании данного типа редакторов.

В число наиболее популярных приложений **WYSIWYG** входит **FrontPage**–приложение, работающее как на платформе **Windows**, так и **Macintosh**. В версии 2000 этого продукта внесены усовершенствования, которые облегчают его использование. Например, код, созданный вручную, не будет перезаписываться в соответствии со спецификациями программы. Внесены улучшения и в интерфейс пользователя. К этой же группе программных продуктов относится программа **GoLive** фирмы **Adobe** и **Dreamweaver** фирмы **Macromedia**. Пользователи **UNIX** могут использовать пакет **WebWorksPablisher** фирмы **Quadralay**.

2.2.4. Шаблоны HTML-документа

Для ускорения работы по созданию **Web**-страниц разработчик может использовать шаблоны, которые видоизменяются в зависимости от содержания страницы. Так, может быть создан шаблон, содержащий фреймы, таблицы, списки и другие элементы **HTML**, необходимые разработчику. Удобно поместить его в папке с

остальными файлами разработки и дать шаблону соответствующее имя, например, **template**.**html**.

Пример шаблона, содержащего основные части **HTML**-документа приведен ниже.

В структуре **HTML**-документа можно выделить следующие части, каждая из которых ограничена парными элементами.

Часть**<html>…….</html>,** внутри которой расположена область *заголовка,* ограниченная элементами **<head>…..</head>**и область *тела* документа между элементами **<body>…….</body>.** Вобластизаголовка**<head>…..</head>**расположены область **<title>……</title>**,содержащая *название (заглавие)* страницы, отображаемое в окне браузера**,** область записи *сценария***<script language="JavaScript">……</script>,** а также область расположения *стилевых спецификаций***<style>…….</style>.**

```
<html>
<head>
<title>
</title>
<script language="JavaScript" type="text/javascript">
<!--
……
//- ->
</script>
<style type="text/css">
……
</style>
</head>
<body>
……
</body>
</html>
```

В **HTML**-документе могут быть использованы *комментарии*, которые заключаются в элементы**<!--…-->.**

В примере 1 приведен листинг **HTML**-документа, который использует элементы создания заголовка первого уровня **<h1>,** абзаца с правосторонним выравниванием **<palign="right">,** логического выделения ****, а также формирования нумерованных **<ol>**и маркированных **<ul>** списков и таблицы **<table>.** В качестве содержимого ячейки таблицы используется графическое изображение **<img>**, а элементами маркированного списка служат гиперссылки **<a>**.

Пример 1

```
<html>
<head>
<title>Пример HTML-кода</title>
</head>
<body text="yellow" bgcolor="#0000f0" link="#00ff00">
<h1 align="center">HTML-документ</h1><br>
<palign="right">Абзац с правосторонним выравниванием</p>
<fontsize=+2 color="#00ff00">Использование<em> логического</em> выделения</font><br>
Нумерованный список:<br>
<ol>
```

```
<li>пункт1</li>
<li>пункт2</li>
<li>пункт3</li>
</ol>
Маркированный список:<br>
<ultype="square">
<li><a href="file1.html">ссылка на файл1</a></li>
<li><a href="file2.html">ссылка на файл2</a></li>
<li><a href="file3.html">ссылка на файл3</a></li>
</ul>
<table border=2 align="center">
<tr><td>cats</td><td><img src="cats.jpg" border=6 ></td></tr>
<tr><td>telephon</td><td>1234567</td></tr>
</body>
</html>
```

Результат отображения кода примера 1 в браузере приведен на рис.2.1.

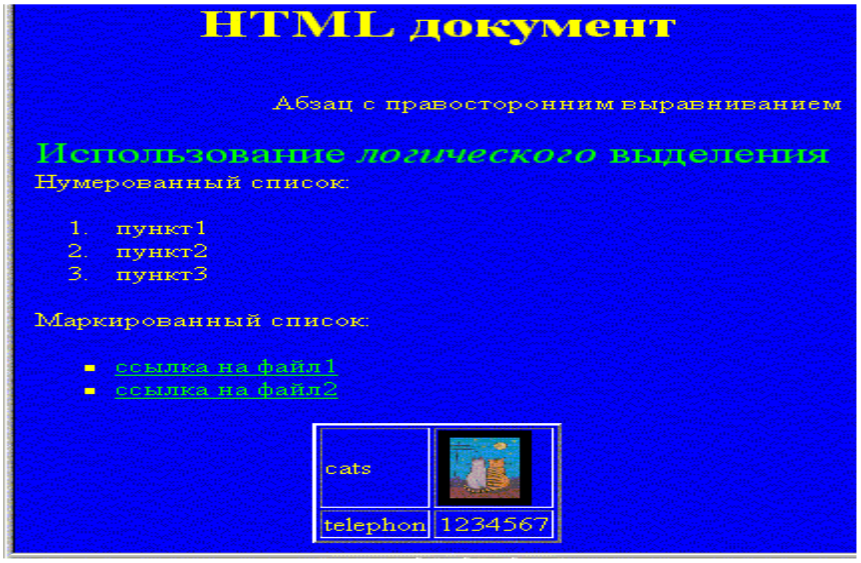


Рис.2.1

2.3. Гиперссылки и навигация

Гиперссылки являются характерным элементом гипертекстовых документов. Рассмотрим элемент HTML, который обеспечивает формирование гиперссылок.

В общем виде гиперссылка на внешний (удаленный) файл представляется как **http://www.server_name/directories/file_name#anh?param**

Ее части: протокол, сервис, имя сервера, директории, ведущие к файлу, якорь, указывающий на фрагмент файла и список параметров, которые необходимо передать серверу образуют **URL (Universal Resource Locator**, универсальный указатель ресурса).

2.3.1. Элемент anchor и его атрибуты

● В **HTML** гиперссылки формируются с помощью элемента **anchor <a>…** и обязательного атрибута **href**- (hyperreference). В качестве контента обычно используется текст:

**текст ссылки **

Часто применяются ссылки, в которых вместо текстового контента используется изображение:

<img src=”file_name.gif”…>

● Кроме обязательного атрибута **href** для пояснения целевой функции гиперссылки применяется атрибут **title**, который формирует всплывающую подсказку при наведении мыши на гиперссылку (рис.2.2)

**text1 **

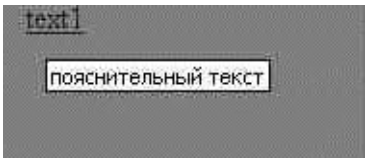


Рис.2.2

● Целевой файл может быть открыт в поименованном окне или в стандартных окнах, имеющих зарезервированные названия. В этих случаях используется атрибут **target**

<ahref=”URL” target=”my_win”>текстссылки

Зарезервированными названиями для окон являются: **_top, _parent, _self, _blank**.

● Для поддержки ссылки клавиатурой используется атрибут **accesskey**

текстссылки

При этом сочетание клавиш **alt+y** активизирует ссылку.

● Можно создавать гиперссылки не только на документы **www**, но и на файлы других сервисов **Internet.**

Ссылка на электронную почту:

Пишите мне

Ссылкана**FTP**:

Архив

Для ссылки на определенный фрагмент страницы используют установку якоря на этот фрагмент, а затем ссылку на якорь.

В пределах одной страницы часто ссылку делают в ее начало, особенно если страница занимает более одного экрана.

Установка якоря производится элементом **<a>** с атрибутом name:

начало текста, а в нижней части страницы устанавливается ссылка на якорь:

в начало текста

2.3.2. Относительные ссылки

Как известно, все файлы сайта должны находиться в одной корневой папке, которая и размещается на сервере.

Внутри сайта используются относительные ссылки на файлы корневой папки.

Пример 1. Создать гиперссылку на файл **image1.gif** из файла **index.html**. Расположение файлов показано на рис.2.3.

В данном случае гиперссылка будет иметь вид:

смотри рисунок

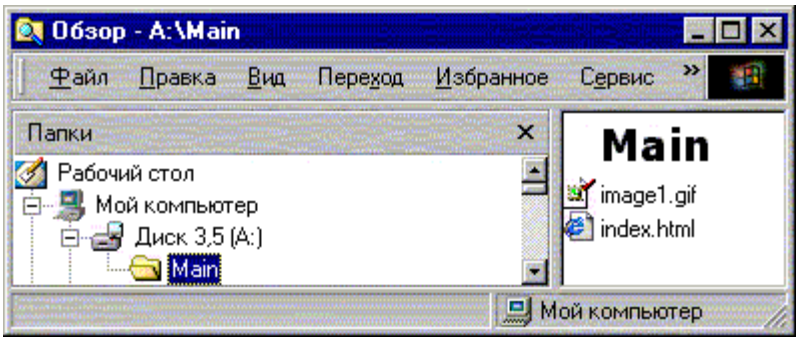


Рис.2.3

Пример 2. Создать гиперссылку на файл **image1.gif** из файла **index.html**. Расположение файлов показано на рис.2.4.

В данном случае гиперссылка будет иметь вид:

смотри рисунок

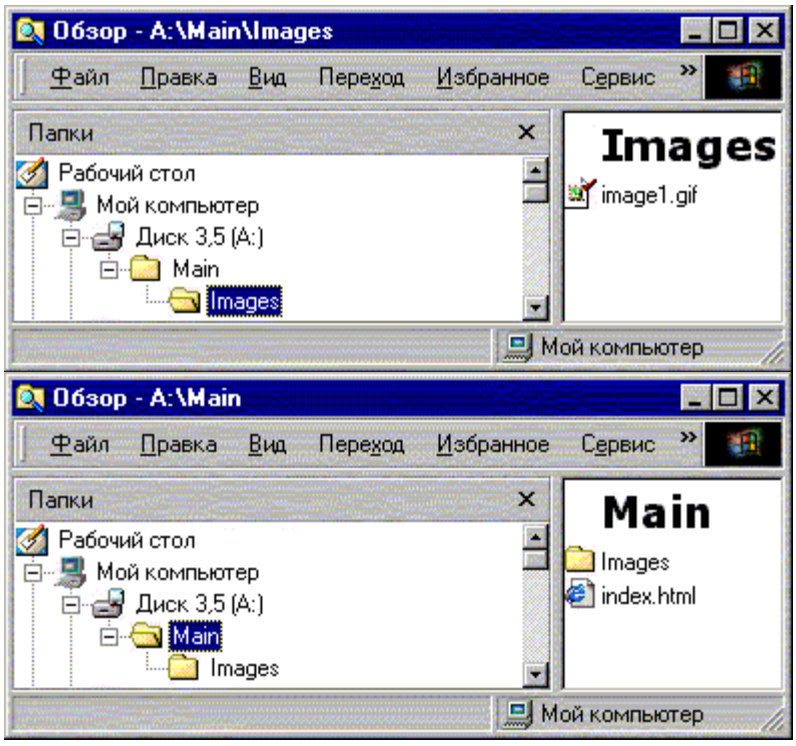


Рис.2.4

Пример 3. Создать гиперссылку на файл **image1.gif** из файла **index.html.** Расположение файлов показано на рис.2.5**.**

В данном случае гиперссылка будет иметь вид**:**

смотри рисунок

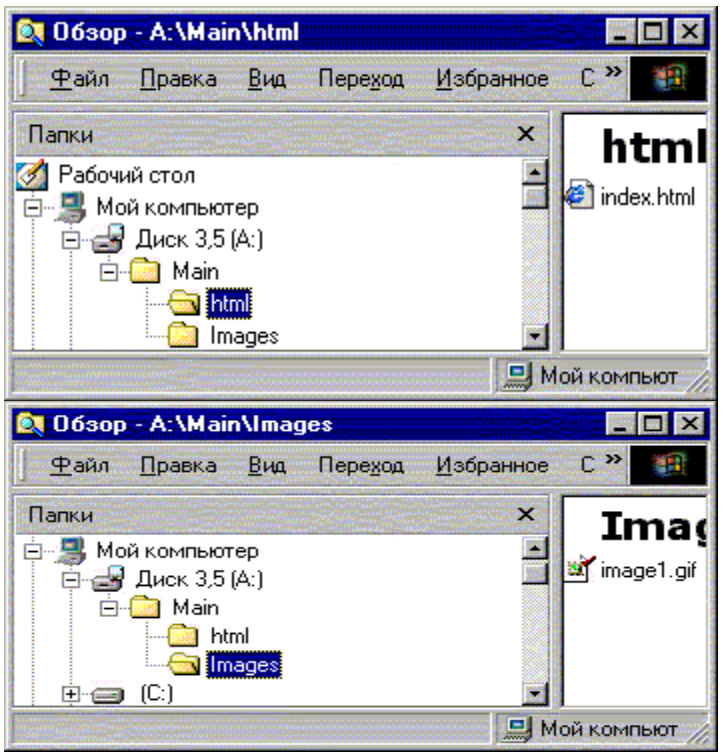


Рис.2.5

2.3.3. Навигация и дизайн

Навигационные элементы должны удовлетворять принципу единства, т.е. иметь одинаковый размер, расположение и оформление на всех страницах. Текстовые ссылки подчиняются тем же правилам, что и графические. Для удобства использования текстовые ссылки зачастую дублируют графические внизу страницы. Навигационные элементы располагают таким образом, чтобы минимизировать движение мыши пользователя. Кроме того, существует неписанное правило о том, что любая информация должна быть получена при использовании не более чем трех щелчков мыши.

Графические кнопки не должны быть громоздкими и вычурно оформленными. Не следует забывать о том, что навигация – служебный элемент сайта.

Принцип динамики диктует расположение элементов навигации сверху или слева, однако на домашней странице навигационная панель может быть расположена и в центре страницы.

2.3.4. Классификации ссылок

Гиперссылки могут быть классифицированы по некоторым признакам.

По *целевому* назначению ссылки разделяются на:

- Внутри страничные;
- Внутри сайта;
- Внешние (на внешние ресурсы).

По *структурированности* ссылки разделяются на:

- Структурированные (образующие навигационные панели или списки);
- Неструктурированные (расположенные в тексте, «оживляющие» страницу и стимулирующие пользователя к тщательному просмотру страницы).

По *динамичности* ссылки разделяются на:

- Статические;
- Динамические (при выборе пользователем какой-либо ссылки, он получает приглашение воспользоваться дополнительными ссылками, которые тематически связаны с уже выбранной… «Вам также понравится…» или «Вас может заинтересовать и этот товар….»)

2.3.5. Виды ссылок

- Текстовые, образуют текстовую панель или используются для дублирования графических ссылок (обычно внизу страницы);
- Графический текст – текст, созданный графическими средствами. Преимуществом является произвольный дизайн, а недостатком – затраты времени на загрузку и ограничение доступа в неграфических средах;
- Графические кнопки, имеющие несколько состояний;
- Пиктограммы (чисто графические или в сочетании с текстом). Они загружаются быстрее графических кнопок, т.к. их размер 6 – 10кб. Вместо текста можно использовать атрибут title. Рекомендуется использовать общепринятый набор пиктограмм с изображением стрелок, дома (переход на домашнюю страницу), письма (переход к электронной почте) и т.д.;
- Карты изображений. Преимуществом является возможность создать горячие области неправильной формы, а недостатком – проблемы в невизуальных средах;
- Рекламные баннеры;
- Выпадающее меню – раскрывающийся список ссылок;
- Меню на основе DHTML – может быть многоуровневым;

- Ролловеры (изменяющееся изображение или текст при наведении и сведении мыши и переход по щелчку);
- Горячие точки, которыми могут стать любые элементы при использовании соответствующей конструкции JavaScript (на примере элемента p):

<ponclick=”location.href=’http://www.yahoo.com’”>переход на yahoo</p>

При этом курсору можно придать вид, характерный для наведения на гиперссылку – руки. Это делается с помощью стилевого описания cursor:hand.

2.3.6. Элементы теории навигации

Навигация – это наука (или искусство) перемещения из одного места в другое. Если географическая навигация имеет ориентиры в виде долготы и широты, отсчитываемых от Гринвича и экватора, то в информационном пространстве такие удобства отсутствуют.

Навигация в **Web** должна помочь пользователям определить свое текущее местоположение, направление, куда им следует двигаться и то, как вернуться в ранее посещенное место.

Цель навигации – добраться до места назначения с наименьшими затратами (времени на щелчки мыши). Хорошая навигация должна снабжать пользователя полной информацией о его местоположении и возможностях дальнейшего передвижения на каждом шаге взаимодействия с сайтом и страницей.

Где я?

Местоположение в **Web** определяет **URL** (**Uniform Resource Locator**). Чтобы улучшить навигацию нужно использовать простые, понятные и запоминающиеся **URL**. **URL** может быть прочитан в адресной строке браузера, иногда его помещают в строку статуса **Web** – страницы. К сожалению, этот ответ на вопрос «Где я?» не всегда может быть полезен или понятен (как в случае динамической генерации страниц), т.к. не содержит никакой информации о связях между страницами.

Точную информацию о своем местонахождении кроме **URL** пользователи могут получить из обозначений страницы. Как правило, в верхней части каждой страницы располагаются ясные обозначения сайта, его раздела и страницы.

Общепринятое соглашение в области дизайна предусматривает расположение в верхнем левом углу страницы логотипа с названием организации, зачастую совпадающие с названием сайта. Логотипы обычно располагают на каждой странице, поскольку они выступают в качестве индикаторов сайта. Для логотипа всегда должно быть определено действие, позволяющее пользователю вернуться на домашнюю страницу сайта. Уместно также сделать подсказку в виде атрибута **title**, указывающего на то, что логотип – это ссылка на домашнюю страницу. Для возвращения на домашнюю страницу также следует использовать более явную возможность, которая заключается в размещении на странице кнопки «Домой».

Еще одним способом указания местоположения является применение обозначения раздела и страницы. Выбранный раздел должен иметь в навигационной панели более блеклое обозначение по сравнению с остальными и не должно быть возможности нажать кнопку, обозначающую раздел. Разделы следующих уровней иерархии также должны быть указаны более блеклым обозначением. Заголовок текущей страницы обычно располагается в ее верхней части и каким-то образом выделяется. Расположение заголовков текущих страниц и их оформление должны быть неизменными на каждой странице.

Иногда на странице используют так называемый.глубиномер, который показывает пользователю, на каком уровне иерархии он находится. Например, пользователь находится на странице **Trainer**

Home| Products| Robots| Trainer

при следующей иерархии:

About

Products----…..

Robots----Butler

Trainer

Security

Friend

News

Jobs

Для указания местоположения иногда применяют цветовое кодирование или темы. В первом случае разделы сайта отличаются цветовым оформлением, а во втором – каждый раздел сайта имеет какое-либо обозначение (чаще всего изображение), характерное для данного раздела. Например, в разделе продукции помещается картинка продавца, в зоне информации для инвесторов – изображение брокера, рассматривающего облигацию и т.д. Недостатком первого метода является излишняя пестрота оформления, а второго – опасность использования метафор, понятных только разработчикам.

Где я был?

Для указания посещенных страниц изменяется цвет ссылки с синего (непосещенная ссылка) на фиолетовый (посещенная). Эти цвета приняты по умолчанию, поэтому менять их не желательно, хотя иногда дизайн сайта требует применения других цветосочетаний.

Помимо изменения цвета ссылок, **Web**-браузер также отслеживает места, где был пользователь, используя журнал. Описание каждой посещенной страницы сохраняется в журнале браузера и пользователь при помощи кнопок Назад и Вперед может переходить по записям в списке. При создании аналогичных кнопок в странице, нужно указывать более точное назначение обратных ссылок, например, Назад к Изделиям.

Вероятно, наиболее передовым способом указания пользователю его последнего посещения является использование cookie. Cookie – это небольшая по объему текстовая информация, сохраненная в системе пользователя сервером. Основной целью использования cookie является сохранение информации об установках пользователя, которые затем могут использоваться для создания комфортного окружения и персонализации, а также предоставить некоторую помощь в навигации.

3. Технологии создания Web-страниц. Каскадные таблицы стилей (CSS)

Технология каскадных таблиц стилей (**CascadingStyleSheets**, **CSS**) была разработана в качестве дополнения к **HTML** с целью определения внешнего вида документов и сохранения за **HTML**только функции структурной разметки документов. Система **CSS** независима от **HTML**, имеет иной синтаксис и позволяет задавать параметры графического (также как и текстового, и звукового) представления дескрипторов **HTML**.

Таблицы стилей – это набор элементов оформления, которые применяются к различным частям документа и описывают способ их представления на экране. В принципе, таблицы стилей реализованы во всех функционально-развитых текстовых процессорах.

С помощью таблиц стилей можно форматировать текст, используя методы, приближенные к методам форматирования обычных печатных страниц. Созданные страницы будут корректно функционировать, учитывая некоторые ограничения, связанные с платформами, браузерами, различными размерами экранов и разрешениями мониторов. Однако процесс разработки **Web**-стандартов в скором времени обещает решить эти проблемы.

Принятие в 1996 году Консорциумом **W3CCSS** первого уровня в качестве стандарта позволило отделить содержание **Web**-страницы (текст, графические изображения и т.д.) от ее оформления (макет страницы и характеристики текста, например, шрифты, цветовое оформление и т.д.). После этого язык **HTML** вновь стал функционально-ориентированным, а не ориентированным на форму. Стандарт **CSS2**, принятый в 1998 году, основан на **CSS** первого уровня и позволяет разработчикам осуществлять контроль над **Web**-страницами на более высоком уровне. Он включает некоторые новые функции, в частности, возможность точно располагать элементы и объекты **Web**-страницы, применять загружаемые шрифты или использовать звуковые таблицы стилей. Применение стилей позволяет также решить многие проблемы поддержки браузеров, т.к. основные компании-разработчики браузеров встроили таблицы **CSS** в свои программные продукты.

Существует три способа применения таблиц стилей в документе **HTML.**

- *Встраивание* (*Inline*) , при котором дескрипторы **Web**-страницы дополняются короткими объявлениями формата. Встраивание предоставляет контроль над фрагментом, к которому оно применяется. Атрибут **style** присоединяется к дескриптору **HTML**, и соответствующий фрагмент страницы будет выводиться браузером с применением формата, указанного стилем.
- *Внедрение* (*Embed*) обеспечивает контроль над страницей **HTML**. Использование дескриптора **<style>** в пределах раздела **<head>** позволяет описать атрибуты стиля, применяемого ко всей странице.
- *Связывание* (*Link*), известное также как применение внешнего листа стилей. В этом случае связанный документ использует эталонную таблицу стилей, применяемую для всего сайта и хранящуюся в файле с расширением .**css.**

Термин "каскадные" (таблицы стилей) описывает в первую очередь тот факт, что браузер следует определенному порядку (каскаду) при интерпретации стилей. Можно использовать все три типа стилей, и браузер интерпретирует их в следующем порядке: связанный – внедренный – встроенный.

3.1. Встраивание стиля

Описание стиля можно встроить в различные элементы **HTML**, для которых стиль имеет смысл, например, в дескрипторы абзацев, заголовков, гиперссылок или ячеек таблицы.

При встраивании в элемент добавляется атрибут **style**, значением которого является свойство и его значение, разделенные двоеточием. Таких пар– "свойство: значение" может быть несколько, в этом случае они разделяются точкой с запятой. Значение атрибута **style** заключается в кавычки.

Два элемента **div**(раздел) и **span**(интервал) являются универсальными элементами-контейнерами и позволяют применять стили к фрагментам текста, как показано в примерах 1 - 3.

Пример 1

```
<p style="font-size: 2em; color: #ff0000">Красный текст с размером шрифта 2ем
</p>
<p style="font-size: 1em; background-color: #ffff00">Текстсразмеромшрифта 1em
    и желтым фоном</p>
```

Пример 2

```
<divstyle="font-size: x-small">Текст с размером шрифта x-small
<span style="font-size: large">Текстсразмеромшрифта large</span></div>
<p style="font-size: 12pt; font-variant: small-caps">Текстсразмеромшрифта
12pt и малыми прописными буквами</p>
<p style="font-size: 16pt; font-family: Garamond, Georgia, serif">Текстс
```

размером шрифта 16pt и серифным шрифтом</p>

Пример 3
<pstyle="font: bolditalic 14ptArial, sans-serif">Текст с сокращенной формой записи свойств шрифта</p>
На рис.3.1 показано отображение в браузере страницы с текстами примеров 1-3.

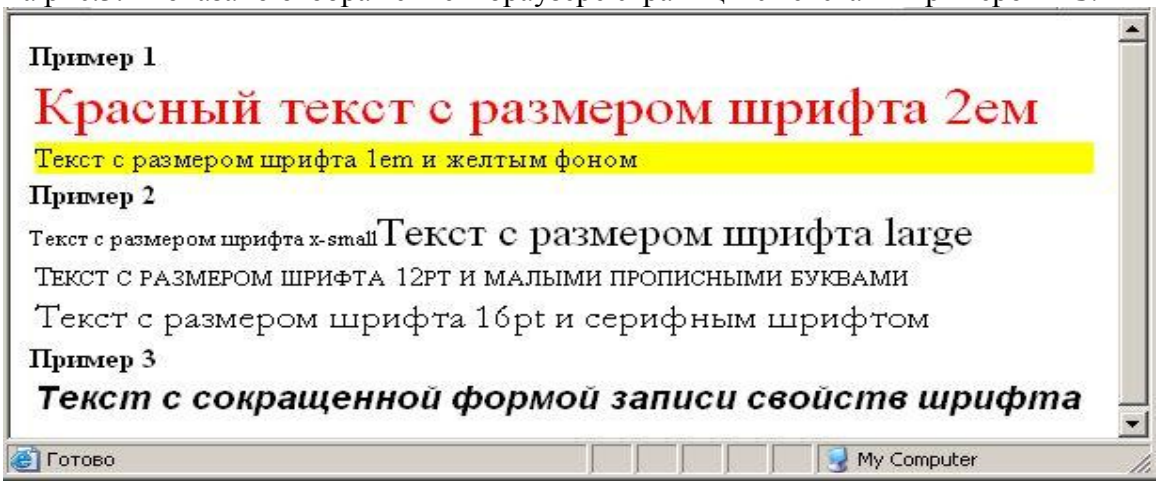


Рис.3.1

3.2. Внедрение стиля

Для внедрения стиля используется элемент (контейнер) **<style>**в пределах раздела **<head>.** Можно скрыть таблицы стилей от просмотра устаревшими браузерами, если содержимое контейнера **<style>**заключить в комментарии (**<!--…-->**), как это сделано в примере 4.

В приведенном далее примере использованы **правила**, по которым должны быть представлены элементы документа. Каждое правило имеет две основные части: **селектор** и **блок объявлений**. Блок объявлений заключается в фигурные скобки и может содержать одно и более объявлений. Каждое объявление представляет собой сочетание **свойства** и **значения**, которые разделяются двоеточием. Внутри блока объявлений может быть задано несколько объявлений, они отделяются друг от друга точкой с запятой.

Пример 4
<html><head>
<title>Пример 4</title>
<style type="text/css">
<!--
body {
background:#000000;color:#ffffff
}
h1 {
font:14pt Verdana; color:yellow
}
-->
</style>
</head>
<body>
**Пример 4

**
**Внедренный стиль для раздела body обеспечивает вывод белых символов на черном фоне
**

<h1>Для вывода заголовка используется внедренный стиль:шрифт Verdana желтого цвета и размером 14 пунктов </h1>

</body></html>

В контейнере **<style>**данного примера находятся описания внедренных стилей. Количество стилей может быть произвольным. Атрибут **type** указывает на используемый тип таблиц стилей - **CSS**.

На рис.3.2 показано отображение текста примера 4 в браузере.

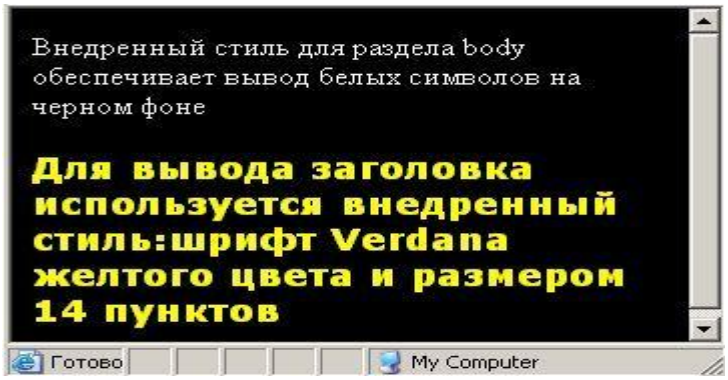


Рис.3.2

3.2.1. Использование различных селекторов для стилевого оформления

3.2.1.1. Селекторы элементов

Селекторы **body** и **h1** примера 4 являются **селекторами элементов** и определяют область применения данного стиля. Селектор **body**указывает на то, что этот стиль будет применен к разделу **body,** а селектор **h1** на то, что стиль будет применен к тексту внутри всех элементов **h1** документа. Таким образом, можно объявить разные стили для заголовков и для остального текста.

В примере 4 парами свойство-значение для селектора **body**являются **background:#000000** и **color:#ffffff**, а для селектора **h1–font:14ptVerdana** и **color:yellow.**

В правиле допускается группирование нескольких селекторов при одном блоке объявлений стиля или же группирование свойств и значений в объявлении стиля.

В примере 5 для того, чтобы придать одинаковые свойства заголовку h5 и абзацу p, используется группирование селекторов.

Пример 5

```
<style type="text/css">
<!--
h5, p  {
font-family:Garamond;
font-size:14pt;
color:#660066;}
-->
</style>
```

В примере 6 показано группирование свойств и значений, относящихся к шрифтовому оформлению.

Пример 6

```
<style type="text/css">
<!--
body  {
font: italic bold 15pt/18pt Verdana,sans-serif;}
-->
</style>
```

При таком группировании должен поддерживаться определенный порядок следования значений свойств, шрифта: наклон (начертание) шрифта, насыщенность шрифта, размер символов, косая черта, межстрочный интервал, семейство шрифтов.

3.2.1.2. Селекторы классов

При необходимости применения правила стилей независимо от элементов используются **селекторы классов**. Классы позволяют также создать несколько наборов стилей на базе одного элемента **HTML**. Название класса предваряется точкой. В примере 7 класс, позволяющий при необходимости применить желтый фон к ячейке таблицы (элемент **td**) имеет селектор вида: **td.yellow**. Класс **black** того же примера не привязан к конкретному элементу и может быть применен к любому элементу текста. В данном примере класс **black** применен для создания фонов ячеек таблицы в шахматном порядке.

Для задания ширины и высоты таблицы применен селектор элемента **table**.

Для применения класса используется атрибут **class.** Применение классов показано в тексте примера 7. Все ячейки, которым присвоен класс стиля **yellow**, будут иметь желтый фон. Аналогично, ячейки, которым присвоен класс **black**, будут иметь черный фон.

Класс **black** применен также и к элементу **p**, текст которого разместится на черном фоне.

Пример7

```
<html><head><title>Пример 7</title>
<style>
td.yellow {background-color:yellow;}
.black {background-color:black;}
table {width:45%;height:30%;}
</style>
</head>

<body>
<strong>Пример 7</strong><br>
<table border=2px>
<tr>
        <td class="black"> </td>
        <td class="yellow"> </td>
        <td class="black"> </td>
</tr>
<tr>
        <td class="yellow"> </td>
        <td class="black"> </td>
        <td class="yellow"> </td>
</tr>
<tr>
        <td class="black"> </td>
        <td class="yellow"> </td>
        <td class="black"> </td>
</tr>
</table>
<pclass="black" style="color:white;text-
align:center">Кабзацупримененклассblackивстроенныйстильвыравниваниясодерж
имогопоцентрусбелымцветомсимволов</p>
</body></html>
```

На рис.3.3 показано отображение текста примера 7 в браузере.

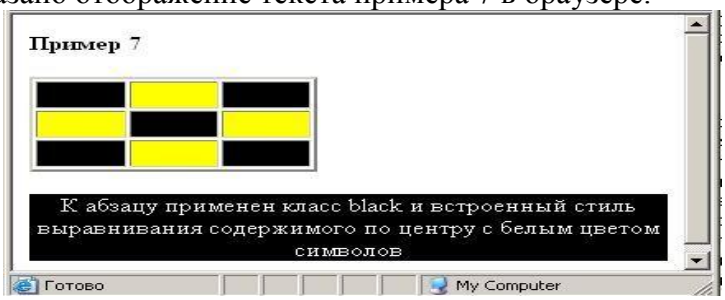


Рис.3.3

3.2.1.3. Селекторы идентификаторов

Селекторы классов можно применять к любому количеству элементов. В правиле стиля можно использовать селектор идентификатора (селектор **ID)**. В отличие от селектора класса, селектор **ID** должен быть уникален для данного документа. Селектор **ID** начинается с символа #. Вместо атрибута **class** в соответствующих элементах **HTML** используется атрибут **id** Его применение иллюстрировано в примере 8. Селектор **#pict** используется для позиционирования графического изображения, которое помещено в контейнер ** .**

Пример 8

```
<html><head><title>Пример 8</title>
<style type="text/css">
#pict {
position:absolute;
top:100;
left:50;}
</style>
</head>

<body>
<strong>Пример 8</strong>
<div id="pict">
<img src="cats.jpg" alt="cats" width=100 height=100>
</div></body></html>
```

Рис.3.4

На рис.3.4 показано отображение текста примера 8 в браузере.

3.2.1.4. Селекторы потомков

Селекторы потомков записываются через пробел после родительского селектора, например, **h1 em {color:red;}**. Это правило сделает красным контент элемента **em**, который является потомком элемента **h1.** В селекторе потомков может быть

использовано несколько элементов, например, **ullia {color:red;}**. Правило применит красный цвет к ссылкам, находящимся в элементах **li** маркированного списка. В примере 19_1 приведены правила с использованием селекторов потомков.

3.2.1.5. Селекторы атрибутов

Селекторы атрибутов могут применяться для выбора элементов на основании их атрибутов и значений этих атрибутов. Селекторы атрибутов были введены в стандарте**CSS2** и уточнялись в стандартах **CSS2.1** и**CSS3**.

Для выбора всех заголовков **h1**, имеющих атрибут **class** и окрашивания текста этих заголовков в серый цвет можно использовать правило **h1[class] {color:gray}** или для выделения полужирным шрифтом текст любой гиперссылки, которая имеет и атрибут **title** и атрибут **href** можно записать **a[title][href] {font-weight: bold}**.

Выбор на основании конкретного атрибута производится следующим образом: **a[title="GotoGUT"] {font-weight: bold}**. Полужирным выделением будут отмкчены гиперссылки, значением атрибута **title**которых будет **GotoGUT**.

Возможен также выбор по частичному значению атрибута, например, **img[title~="Рисунок"]{border:1pxsolidgray}**. Это правило обеспечит нарисование рамки вокруг изображений, в атрибуте **title** которых содержится слово "Рисунок".

3.2.1.6. Псевдоклассы и псевдоэлементы

В **CSS2.1** определены псевдоклассы и псевдоэлементы. Традиционно псевдоклассы используются для оформления гиперссылок. Они описывают свойства ссылок непосещенных (**:link**), посещенных (**:visited**), над которыми находится курсор мыши (т.е. свойства ссылок "при наведении" курсора мыши) (**:hover**) и активных ссылок (**:active**). В примере 9 приведен код **HTML**, применяющий псевдоклассы для элемента **<a>**. В данном примере непосещенные ссылки имеют черный цвет, активные ссылки окрашиваются в синий, посещенные - в зеленый, а при наведении мыши на ссылку цвет ее меняется на красный и размер шрифта увеличивается до 16 пунктов (рис.3.5). Порядок расположения псевдоклассов в селекторе важен, так как при его нарушении принцип каскадности применения стилей может привести к игнорированию свойств отдельных псевдоклассов.

Пример 9

```
<html><head><title>Пример 9</title>
<style type="text/css">
<!--
a:link  {color:black;}
a:visited {color:#00ff00;}
a:hover  {color:#ff0000;font-size:16pt;}
a:active  {color:#0000ff;}
-->
</style>
</head>

<body>
<strong>Пример 9</strong><br>
<a href="классы_1.html" >Переходкфайлу "классы_1.html"</a><br>
<a href="классы.html" >Переходкфайлу "классы.html"</a>
</body></html>
```

На рис.3.5 показано отображение в браузере результата действия псевдокласса :hover - наведения мыши на вторую гиперссылку.

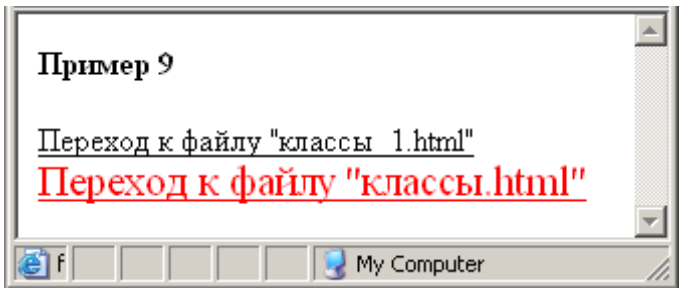


Рис.3.5

К псевдоклассам относится также**:focus**. Этот псевдокласс относится к любому элементу, которому в настоящий момент принадлежит фокус ввода, т.е. который готов принимать ввод с клавиатуры или быть активированным другим способом. (**IE7.0** не поддерживает этот псевдокласс для элементов формы).

Псевдоэлементы вводят фиктивные элементы в документ, чтобы достигнуть определенных эффектов. В **CSS2.1** определены четыре псевдоэлемента, для которых могут быть применены специальные стили: первая буква (**:first-letter**), первая строка (**:first-line**), до (**:before**) и после (**:after**) элемента.

Например, правило **h3:first-letter {font-size:150%}** увеличит первую букву заголовка в полтора раза по отношению к остальному тексту. А правило **p:first-line {color:red}** сделает символы первой строки абзаца красными.

Примеры 10 и 11 иллюстрируют использование псевдоэлементов**:first-letter**и **:first-line.**

Пример 10

```
<html><head>        <title>Пример 10</title>
<style type="text/css">
p:first-letter{
font-size:3em;
font-weight:bold;
color:#ff0000;}
p {line-height:1em;text-align:justify;}
</style>
</head>

<body>
<strong>Пример 10</strong>
<p>Псевдоэлементfirst-letterиспользуетсядлясозданиябуквицывабзаце.
Абзацимеетбуквицуразмера 3еминачертанияbold.</p>
<p style="text-indent:200px">Абзац имеет буквицу размера 3 ем, начертания
bold  и отступ первой строки 200px. Абзац имеет буквицу размера 3 ем, начертания
bold  и отступ первой строки 200px.</p>
</body></html>
```

На рисунке 3.6 показано отображение примера 10 в браузере.

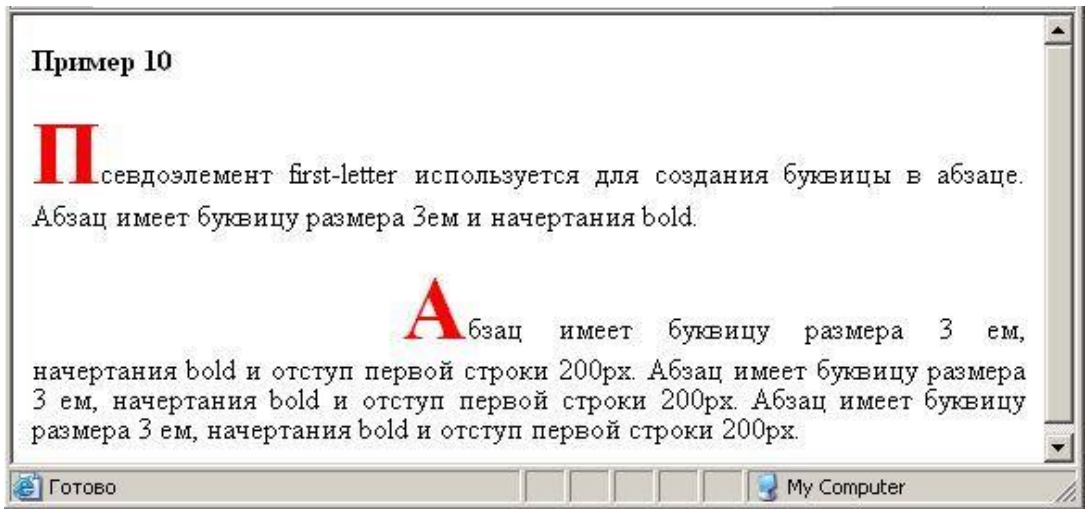


Рис.3.6

Пример 11

```
<html><head>        <title>Пример 11</title>
<style type="text/css">
p:first-line{
font-weight:bold;}
p {line-height:1em;text-align:justify}
</style>
</head>

<body>
<strong>Пример 11</strong>
<p>Первыестрокиабзацевстилизованыспомощьюпсевдоэлементар:first-line.
Первые строки абзацев стилизованы с помощью псевдокласса p:first-line.  </p>
<p > Первые строки абзацев стилизованы с помощью псевдоэлемента p:first-
line. Первые строки абзацев стилизованы с помощью псевдокласса p:first-line. </p>
</body></html>
```

На рисунке 3.7 показано отображение примера 11 в браузере.

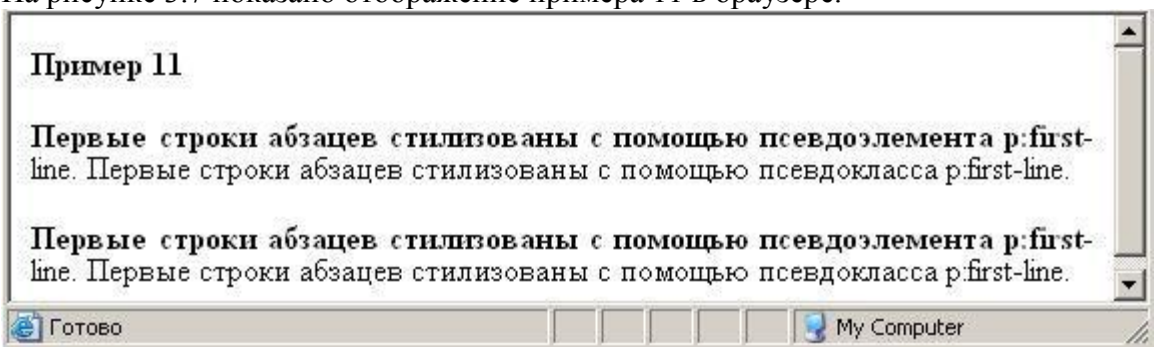


Рис.3.7

Псевдоэлементы**:before**и **:after**позволяют вставлять *генерируемое содержимое* и затем применять к нему специальные стили. Генерируемое содержимое вставляется с помощью свойства **content**, например, **h2:before {content: "ГУТ"}** добавляет перед заголовками второго уровня текст "ГУТ" илидля **body:after {content: "Theend"}** добавляет в конец документа текст "Theend".

3.3. Связывание стиля

При связывании стиля все стилевые описания производятся по тем же правилам, что и при внедрении, но хранятся они в отдельном файле, имеющем расширение .**css**.

Сам файл должен находиться в корневом каталоге сайта, в противном случае нужно корректно указать связь с ним.

Преимуществом связывания является то, что стилевые описания применяются ко всем страницам сайта и при необходимости изменения оформления достаточно внести новшества в файл, содержащий эти описания, вместо того, чтобы исправлять все страницы сайта.

В приведенном ниже примере иллюстрируется использование внешней таблицы стилей.

Пример

Файл с именем **my_style.css** содержит *только* правила стилей:

```
body  {
background: #000000;
color: #ffffff;
}
a  {
color: #ff0000; text-decoration:none;
}
```

Документ **HTML**, который ссылается на этот файл, должен содержать в части **head** следующую ссылку, использующую элемент **link**:

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

Аналогом элемента **link** является директива **@importurl (url)**. Директива **@import** также указывает браузеру на необходимость загрузки внешней таблицы стилей. Размещается директива в контейнере **<style>** перед остальными правилами **CSS**. Примеры использования директив **@import** приведены далее:

```
<style>
@import url ("sheets22.css");
@import url ("http://example.org/library/layout.css");
@import url ("printer.css") print;
</style>
```

Из примера видно, что может быть использовано несколько директив **@import** для присоединения нескольких файлов с таблицами стилей. В последней директиве указано устройство, для которого создана присоединенная таблица.

3.4. Текстовые свойства CSS

1. text-align - выравнивание текста со значениями: **left** (по левому краю), **right** (по правому краю), **center** (по центру), **justify** (по обоим краям).

2. text-indent - отступ в первой строке блока (абзаца, раздела) со стандартными значениями длины (**pt, px, cm, mm**).

3. text-decoration - оформление текста подчеркиванием со значениями: **none** (отсутствует - по умолчанию), **underline** (подчеркивание), **overline** (линия над текстом), **line-through** (перечеркивание), **blink** (мерцание).

4. text-transform - перевод букв в верхний или нижний регистр со значениями: **none** (отсутствует - по умолчанию), **capitalize** (первая буква каждого слова становится прописной), **uppercase (**переводит все буквы в верхний регистр), **lowercase** (переводит все буквы в нижний регистр).

5. text-shadow - установка эффекта затенения текста со значениями: **none** (отсутствует - по умолчанию), **colorlefttopradius** (цвет затенения с расстояниями слева (справа), вниз (вверх) от текста и радиусом нерезкости).

6. letter-spacing - расстояние между символами текста со стандартными значениями длины (**pt, px, cm, mm, em,ex**).

7. word-spacing - расстояние между словами со стандартными значениями длины **(pt, px, cm, mm**).

Кроме стандартных единиц измерения длины **px, pt** (равный 0,35мм), **mm, cm**, может использоваться **em** (**1em** равен ширине буквы m в шрифте заданного размера), **ex** (**1ex** равен высоте шрифта), **px** (**1px** равен ширине пиксела).

Пример 12

```
<html><head><title>Пример 12</title>
<style type="text/css">
<!--
p.class1 {
text-align: justify;
text-indent: 20pt;
color: #c0c0c0;
background-color: #000000;
}
.class2  {
text-decoration: underline;
}
.class3 {
letter-spacing: 0.5em;
}
.class4  {
text-transform: uppercase;
}
-->
</style>
</head>

<body>
<strong>Пример 12<b/>
<pclass="class1">Текст абзаца выравнивается по ширине, имеет отступ первой строки 20 пунктов и использует белый цвет символов на черном фоне.</p>
<p>В тексте абзаца <spanclass="class2">используется подчеркивание, </span>
<spanclass="class3"> увеличение расстояния между символами</span>
<spanclass="class4">, а также перевод символов в верхний регистр.</span></p>
</body></html>
```

На рис. 3.8 показано отображение текста примера 12 в браузере.

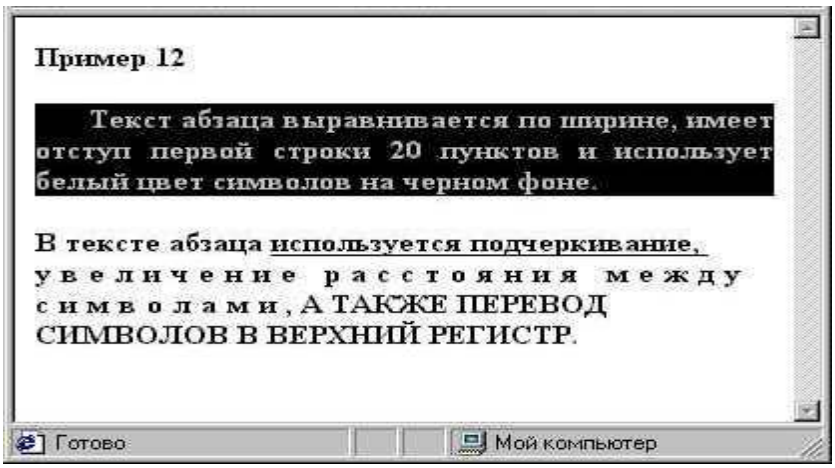


Рис.3.8

3.5. Цвет и фон

1. color - цвет текста - любое значение, соответствующее стандарту.

2. background-color - цвет фона - любое значение, соответствующее стандарту.

3. background-image:URL("URL") - фоновое изображение - любое значение **URL**, соответствующее стандарту.

4. background-repeat- направление повторения фонового изображения со значениями: **repeat** (повторение по горизонтали и вертикали - по умолчанию), **repeat-x** (повторение только по горизонтали), **repeat-y** (повторение только по вертикали), **no-repeat** (отсутствие повторения).

5. background-position - положение фонового изображения относительно верхнего левого угла содержащего его элемента со значениями: расстояние по горизонтали, расстояние по вертикали в стандартных единицах длины или в процентном соотношении. Значение 50% по горизонтали и 50% по вертикали дает расположение фонового изображения по центру. Размещение фонового изображения может быть также **top (**по верхнему краю),**center** (по центру), **bottom**(по нижнему краю), **left** (по левому) и **right**(правому) краям.

6. background-attachment - прокрутка фонового изображения вместе с документом со значениями: **scroll** (прокрутка - по умолчанию) или **fixed** (фиксация) изображения в окне браузера.

7. Для описания сразу всех параметров фона можно использовать свойство **background: background-colorbackground-imagebackground-repeatbackground-attachmentbackground-position**

Пример 13_1

```
<html><head><title>Пример 13_1 </title>
<style>
p {background-image: url(images3.jpg);
background-position:center;
border: 1px dotted gray;}
p.c1 {background-repeat: repeat-y;}
p.c2 {background-repeat: repeat-x;}
</style>
</head>

   <body><strong>Пример 13_1</strong>

<pclass="c1"> В данном абзаце фоновое изображение позиционировано по центру и распространяется по вертикали. В данном абзаце фоновое изображение позиционировано по центру и распространяется по вертикали. </p>
<pclass="c2"> В данном абзаце фоновое изображение позиционировано по центру и распространяется по горизонтали. В данном абзаце фоновое изображение позиционировано по центру и распространяется по горизонтали.</p>
</body></html>
```

На рис.3.9 показано отображение текста примера 13_1 в браузере.

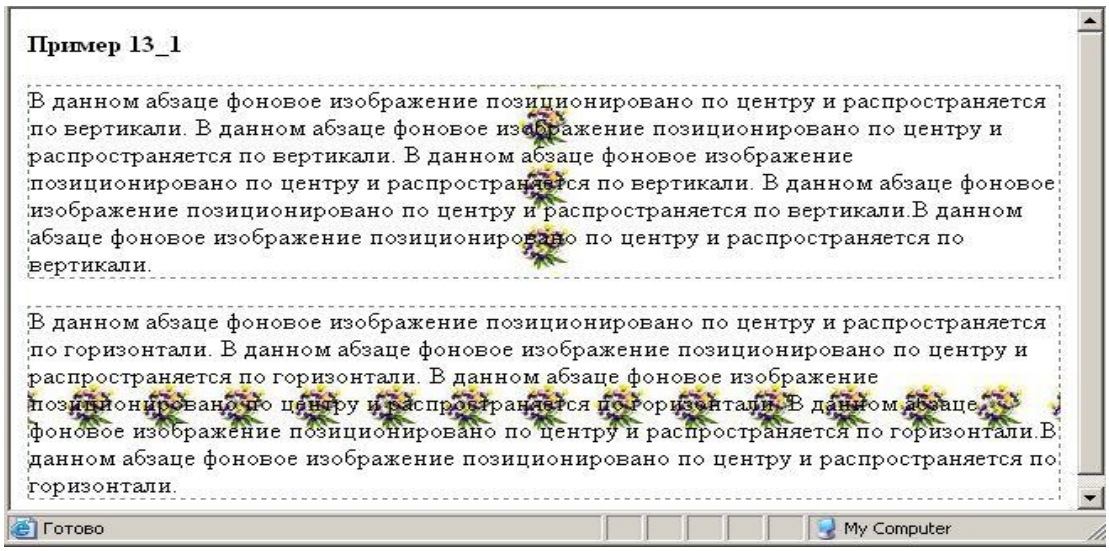


Рис.3.9

В примере 13_2 фоновое изображение позиционировано на 25% по горизонтали и, по умолчанию, по центру по вертикали.

Пример 13_2

```
<html><head><title>Пример 13_2</title>
<style>
p {background-image: url(images3.jpg);
background-position:25%;
background-repeat: no-repeat;
border: 2px dotted gray;}
</style>
</head>
<body>
<strong>Пример 13_2</strong><br>
<p>В данном абзаце фоновое изображение позиционировано на 25% по горизонтали и по центру по вертикали. В данном абзаце фоновое изображение позиционировано на 25% по горизонтали и по центру по вертикали. </p>
</body></html>
```

На рис.3.10 показано отображение текста примера 13_2 в браузере.

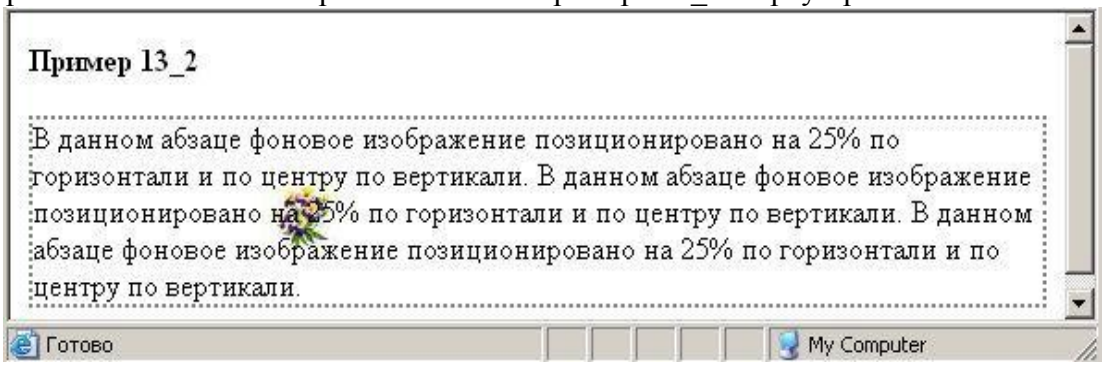


Рис.3.10

3.6. Шрифты

1. font-family - семейство шрифтов. Может быть несколько семейств, отделенных друг от друга запятыми. Приоритет определяется порядком в списке. Значением может быть имя типового шрифта (**serif/sans-serif/cursive/fantasy/monospace**), а также шрифта, принадлежащего одному из типов.

2. font-style - начертание шрифта со значениями: **normal (**обычноепо умолчанию),–, курсив (**italic)**, наклонное начертание (**oblique).**

3. font-variant - в виде малых прописных букв со значениями: **normal** (обычное - по умолчанию) или **small-caps** (в виде малых прописных букв).

4. font-weight - толщина выводимого шрифта со значениями: **normal** (обычная - по умолчанию), **bold** (полужирный), **bolder** (жирный), **lighter** (светлый) или числовыми значениями: 100-светлый, 400-обычный, 700-полужирный, 900-жирный.

5.font-size - высота символов (кегль). Значением является любая, соответствующая стандартам высота или процентное значение, обозначающее уменьшение или увеличение в процентах от кегля родительского элемента. Значения абсолютного размера могут быть записаны в виде ключевых слов: **xx-small** (крайне малый), **small** (малый), **medium** (средний -по умолчанию), **large** (большой), **x-large** (очень большой), **xx-large** (крайне большой).

Значения относительного размера могут быть записаны в виде ключевых слов: **larger** (больше) и **smaller** (меньше).

6. Для описания сразу всех параметров шрифта можно использовать свойство **font: font-stylefont-variantfont-weightfont-sizefont-family**

Использование различных свойств шрифтов приведено в примере 14.

Пример 14

```
<html><head><title>Пример 14 </title>
<style type="text/css">
p {
font: oblique  18pt Impact;}
.p1 {font: large Verdana, sans serif;
color:#0000ff;}
</style>

</head>

<body>
<strong>Пример14</strong>
<p>Для абзаца применен шрифт Impact стиля oblique, 18 пунктов</p>
<p class="p1">Для абзаца применен шрифт Verdana, large</p>
</body></html>
```

На рис.3.11 показано отображение текста примера 14 в браузере.

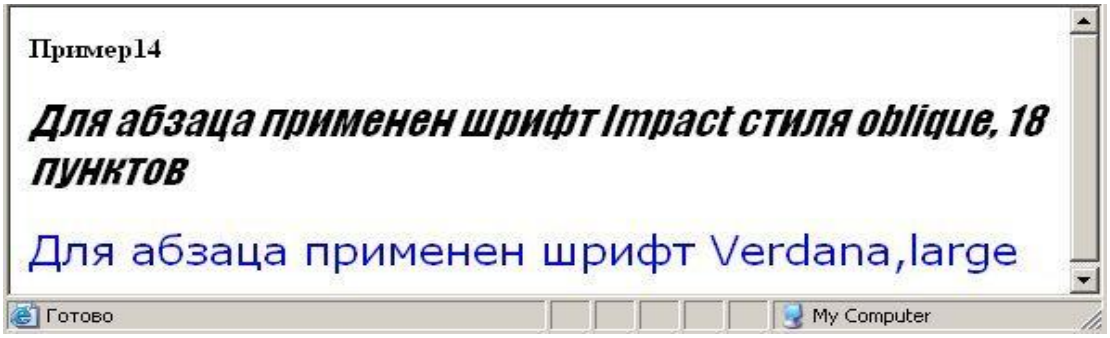


Рис.3.11

3.7. Блочная модель

1. margin-top, margin-right, margin-bottom, margin-left–ширинаверхнего, правого, нижнегоилевогополя. Значение по умолчанию – 0, значениями являются длины, соответствующие стандарту или процентное значение, определяющее отношение ширины поля к ширине элемента.

2. margin - ширина полей для всех сторон элемента. У этого свойства может быть от одного до четырех значений. Одно значение присваивается всем полям, из двух значений первое присваивается *верхнему* и *нижнему* полю, а второе – *левому* и *правому*. При трех: первое -верхнему полю, второе – левому и правому, а третье – нижнему.

3. padding-top, padding-right, padding-bottom, padding-left - ширина промежутка между содержимым элемента и определенным участком его границы. Значение по умолчанию – 0, значениями являются длины, соответствующие стандарту или процентное значение, определяющее отношение ширины промежутка к ширине элемента.

4. padding - ширина промежутка для всех сторон элемента. У этого свойства может быть от одного до четырех значений. Одно значение присваивается всем промежуткам, из двух значений первое присваивается *верхнему* и *нижнему* промежутку, а второе – *левому* и *правому*. При трех: первое - верхнему полю, второе – левому и правому, а третье – нижнему.

5. border – рамка. Для описания всех свойств рамки можно использовать это конструкцию:

border: border-widthborder-styleborder-color.

Свойствами являются **border-width (**ширина рамки), **border-style (**стиль рамки) и **border-color** (цвет рамки).

5.1. border-width - ширина рамки. Значениями являются: **thin (**тонкая линия), **medium (**средняя – по умолчанию), **thick (**толстая), а также стандартные значения ширины.

Можно также устанавливать значения ширины рамки для определенной стороны. Для этого используются свойства:

border-top-width, border-right-width, border-bottom-width, border-left-widthc аналогичными значениями**.**

5.2 border-style–стиль рамки. Значениями являются: **none (**отсутствие – поумолчанию), **hidden (**скрытая), **dotted (**пунктир), **solid (**сплошная), **double (**двойная), **dashed (**штрих-пунктир), **groove**(двойная борозда), **ridge** (гребень), **inset (**врезка**), outset (**орнамент).

Можно также устанавливать значения стиля рамки для определенной стороны. Для этого используются свойства:

border-top-style, border-right-style, border-bottom-style, border-left-style.

5.3.border-color - цвет рамки. Значением является любое, соответствующее стандарту.

Можно также устанавливать значения цвета рамки для определенной стороны. Для этого используютсясвойства:

border-top-color, border-right-color, border-bottom-color, border-left-color

Пример 15

```
<html><head><title>Пример 15</title>
<style type="text/css">
p  {
border: 10px double red;
margin: 50px 100px 100px 50px;
padding:20px }

</style>
```

</head>
<body>
Пример15
** **Текст абзаца помещен в двойную рамку красного цвета, толщиной 10 пикселей. Установлены поля шириной 100 пикселей справа и снизу и 50 пикселей сверху и слева, а промежуток между текстом абзаца и рамкой установлен 20 пикселей со всех сторон.
</body></html>

На рис.3.12 показано отображение текста примера 15 в браузере.

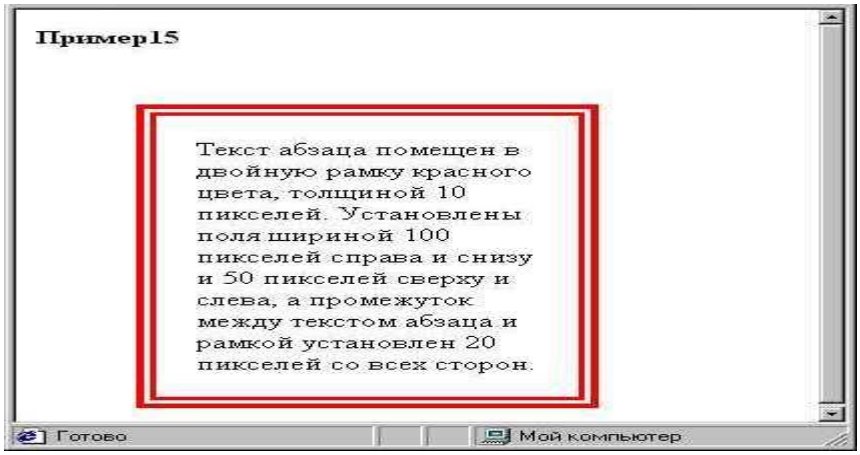


Рис.3.12

3.8. Плавающая модель для размещения изображений и других элементов

До появления последних стандартов **CSS** для обтекания текстом изображений пользовались атрибутом **align** элемента **img** со значениями **left** и **right**. В настоящее время стандартизовано свойство **float** со значениями **left** и **right**, которое может быть применено не только к изображениям, но и к другим элементам **HTML**. В сочетании со свойством **float** часто применяют свойство **clear** со значениями **left** и **right** или **both** для того, чтобы освободить место слева или (и) справа от других элементов Web-страницы.

В примере 16_1 показано использование плавающей модели для изображения.

Пример 16_1

```
<html><head>        <title>Пример 16_1</title>
        <style>
        .leftFloat {float:left}
        .rightFloat {float:right}
        </style>
</head>

<body>
<strong>Пример 16_1</strong><br>
<img src="flax.gif" class="leftFloat" alt="flax" width="80px" height="80px" border="1px">
<p>Для обтекания текстом изображения слева  вместо атрибута align тега img используется свойство float со значением left. Для обтекания текстом изображения слева  атрибута align тега img используется свойство float со значением left. Для обтекания текстом изображения слева  вместо атрибута align тега img используется свойство float со значением left. </p>
```

<img src="flax.gif" class="rightFloat" alt="flax" width="80px" height="80px" border="1px">

** Для обтекания текстом изображения справа вместо атрибута align тега img используется свойство float со значением right. Для обтекания текстом изображения справа атрибута align тега img используется свойство float со значением right. Для обтекания текстом изображения справа атрибута align тега img используется свойство float со значением right. **

</body></html>

На рис.3.13 показано отображение текста примера 16_1 в браузере.

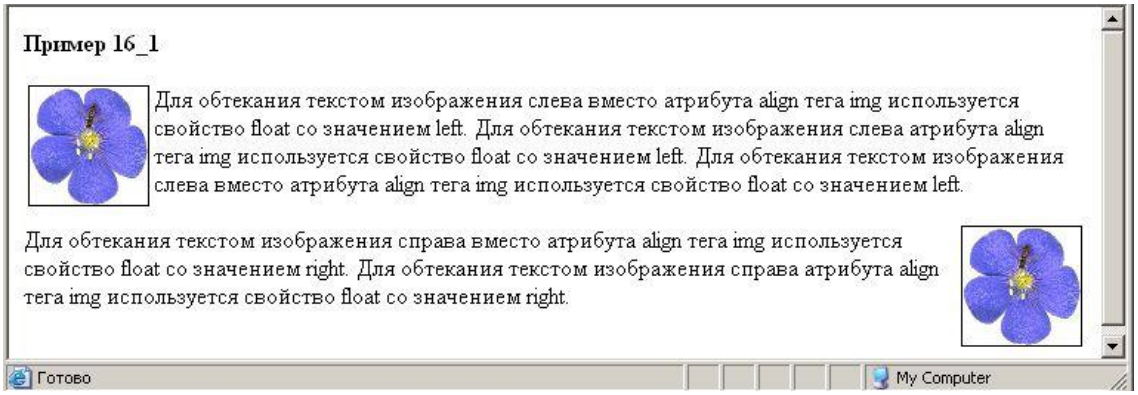


Рис.3.13

В примере 16_2 показано использование плавающей модели для абзаца с цитатой.

Пример 16_2

```
<html><head><title>Пример 16_2</title>
<style>
p {font-family:Garamond;
text-align:justify;
margin-left:0.25em;}
.clearp {clear:left;}

 p.Floatl {float:left;
width:45%;
text-align:center;
margin:0.5em 0.25em;
padding:0.25em;
border-top:1.3em  solid #999 ;
border-bottom:1.3em  solid #999 ;
background-color:black;
color:white;
font-size:1.3em;
font-family:Garamond;
}
</style>
</head>

<body>
<strong>Пример 16_2</strong>
<p>В примере используется обтекание текстом абзаца цитаты. Свойства цитаты записаны в классе .Floatl</p>
<p  class="Floatl">"Фантазия важнее знаний" А.Эйнштейн </p>
```

** **Обтекание цитаты, заключенной в абзац, обеспечивается свойством float. Для обтекаемого элемента должна быть обязательно задана его ширина, кроме того в примере указаны свойства margin и padding, рамки серого цвета сверху и снизу, а также свойства шрифта: семейство, цвет символов, цвет фона и размер. Обтекание цитаты, заключенной в абзац, обеспечивается свойством float. Для обтекаемого элемента должна быть обязательно задана его ширина, кроме того в примере указаны свойства margin и padding, рамки серого цвета сверху и снизу, а также свойства шрифта: семейство, цвет символов, цвет фона и размер.

</body></html>

На рис.3.14 показано отображение текста примера 16_2 в браузере.

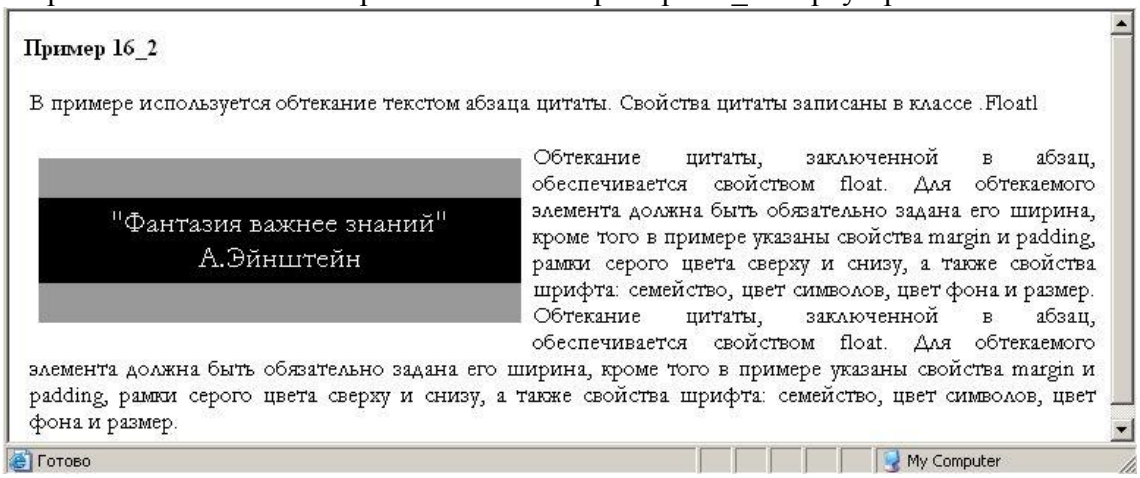


Рис.3.14

В примере 16_3 показано использование обтекания для создания трехколонного документа с вставленной в одну из колонок цитатой.

Пример 16_3

```
<html><head><title>Пример 16_3</title>
<style type="text/css">
#header {color: white;
background-color: #666;
border-bottom:1px solid #333;
text-align: center;
padding: 0.1em 0;
margin: 0 0 1em 0}

#leftcol {width: 25%;
float:left;
background: white;
padding-bottom: 1em;
text-align: left;
}

#rightcol {
width:30%;
float: left;
background: white;
padding-left: 1em;
}

#centercol {
```

```
width:43%;
float: left;
background: white;
padding: 0 1em;
border-left: 1px solid #333;
border-right: 1px solid #333
}
#cite {
float:right;
width:90%;
border: 1px solid red;
padding:0.1em;
margin: 0 0.1em;
font-size:larger;
text-align:center;
}
#footer {
clear: both;
padding-bottom: 1em;
border-top: 1px solid #333;
text-align: center;}
</style>
</head>
<body>

<div id="header">
<h3>Технологиякаскадныхтаблицстилей (Cascading Style Sheets, CSS)</h3>
</div>
<div id="leftcol">
<p>Таблицы стилей - это набор элементов оформления, которые применяются к различным частям документа и описывают способ их представления на экране. В принципе, таблицы стилей реализованы во всех функционально-развитых текстовых процессорах.</p></div>
<div id="centercol">
<p>Применение стилей позволяет также решить многие проблемы поддержки браузеров, т.к. основные компании-разработчики браузеров встроили таблицы CSS в свои программные продукты.
</p></div>
<div id="rightcol">
<p>С помощью таблиц стилей можно форматировать текст, используя методы, приближенные к методам форматирования обычных печатных страниц. Созданные страницы будут корректно функционировать, учитывая некоторые ограничения, связанные с платформами, браузерами, различными размерами экранов и разрешениями мониторов.  Процесс разработки Web-стандартов в скором времени обещает решить эти проблемы.</p>
<p id="cite">"Искусство - это отражение мира в образах, а наука - отражение мира в понятиях" Н. К. </p></div>
<div id="footer">Консорциум W3C CSS</div>
</body></html>
```

На рис.3.15 показано отображение текста примера 16_3 в браузере.

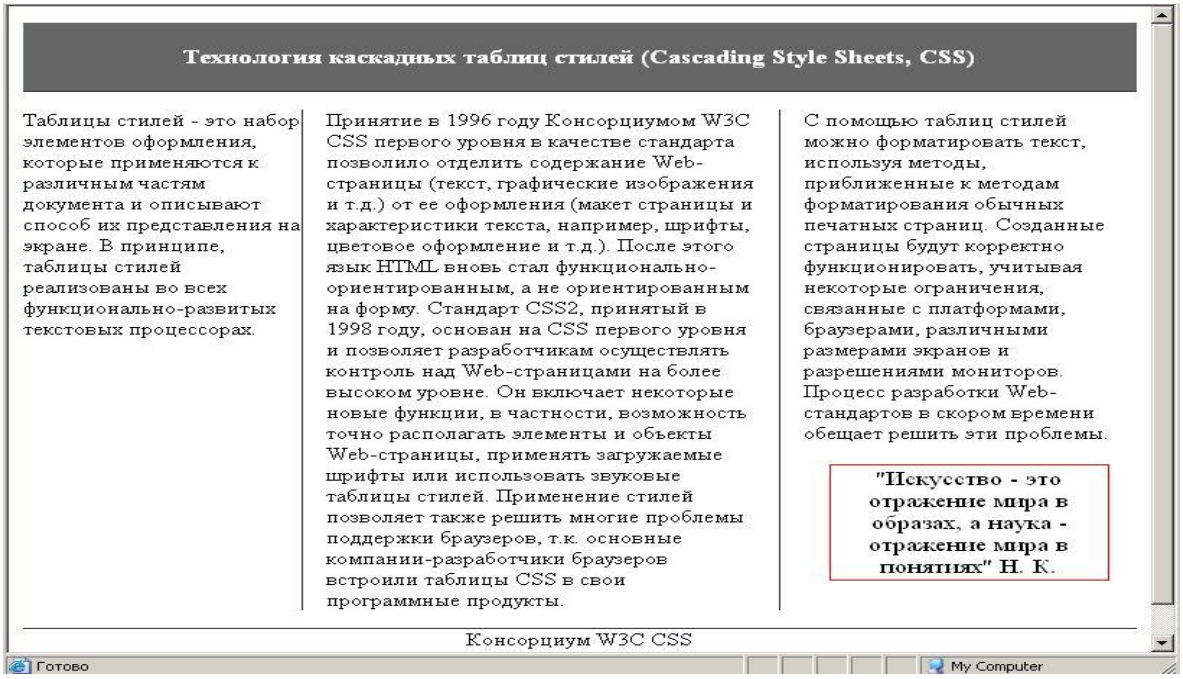


Рис.3.15

3.9. Позиционирование и визуализация

Применение позиционирования предполагает использование ряда понятий:

Ограниченная область (блок-контейнер) – невидимая прямоугольная область, определенная браузером. Таблицы стилей позволяют управлять этой областью, устанавливая ее положение на странице с использованием абсолютных или относительных значений позиционирования.

Абсолютное позиционирование – технология, позволяющая задавать координаты блока относительно его блока–контейнера, которым может быть другой (родительский) элемент документа или начальный блок–контейнер – прямоугольник, соответствующий окну просмотра браузера. При этом блок элемента полностью удаляется из потока документа и позиционируется относительно его блока–контейнера.

Относительное позиционирование - технология, позволяющая задавать координаты блока относительно его несмещенного положения в потоке документа. При относительном позиционировании элемент сдвигается со своего обычного места, но пространство, которое он должен был занимать, не исчезает.

Фиксированное позиционирование – технология, позволяющая задавать координаты блока относительно начального блока – контейнера (окна просмотра браузера).

1. position - метод позиционирования блока, по умолчанию имеет значение **static**, другими значениями являются **absolute** (абсолютное) и **relative** (относительное) позиционирование, а также значение – **fixed**, при котором позиционирование блока является смещением, как в случае абсолютного позиционирования, но блок фиксируется в окне браузера и не перемещается при прокрутке окна.

2. top - величина смещения вниз от верха его блока – контейнера, **bottom** – вверх от нижней стороны блока - контейнера, **left** – соответственно, влево, а **right** – вправо от левой и правой сторон блока - контейнера. Значениями являются любые, соответствующие стандарту длины, а также процентное значение: отношение в процентах длины смещения к ширине (высоте) блока, а также допустимы отрицательные значения указанных свойств смещения.

3. width - ширина блока. Значениями являются любые, соответствующие стандарту длины, а также процентное значение: отношение в процентах длины смещения к ширине окна.

4. height - высота блока. Значениями являются любые, соответствующие стандарту длины, а также процентное значение: отношение в процентах длины смещения к высоте окна.

5. z-index - z-индекс определяет порядок расположения блоков. Значениями являются целые числа (положительные и отрицательные), причем блоки с большими значениями z-индекса будут появляться над блоками с меньшими значениями.

6. visibility – видимость. Определяет, является ли элемент видимым - **visible** или скрытым - **hidden**.

7. overflow - управление переполнением. Имеет три значения: **visible** (элемент виден), **hidden** (перекрываемая часть отсекается), **scroll** (используется механизм прокрутки для визуализации элемента).

8. clip: rect (top right bottom left) – отсекание. Определяет вырезаемые области. Вырезаемая область определяется значениями сдвига соответственно сверху, справа, снизу и слева.

9. **display** – представление элемента со значениями **none**, **inline, block, list-item, table, inline-table** и другие значения, не рассматриваемые в данном пособии. Дополнительные сведения можно получить на сайте Консорциума WWW**http://www.w3.org/TR/CSS21/**

Пример 17_1 иллюстрирует основные варианты позиционирования.

```
<html><head><title>Пример 17_1</title>
<style  type="text/css">
<!--
p, h5 {margin:0;padding:0}
-->
</style>
</head>

<body>
<strong>Пример 17_1</strong>
<div style="background-
color:'#c0c0c0';position:absolute;top:70;left:50;color:'#0000cc';">
<h5>relative</h5>
<p>Test text test text</p>
<p style="position:relative;top:70;left:50;color:'#cc0000'">
Test text test text</p>
</div>

<div style="background-
color:'#c0c0c0';position:absolute;top:70;left:250;color:'#0000cc'">
<h5>normal stream</h5>
<p>Test text test text</p>
<p style="color:'#cc0000'">Test text test text</p>
</div>

<div style="background-
color:'#c0c0c0';position:absolute;top:70;left:450;color:'#0000cc';>
<h5>absolute</h5>
<p>Test text test text</p>
<p style="position:absolute;top:70;left:50;color:'#cc0000'">Test text test text</p>
</div>
</body></html>
```

На рис.3.16 показано отображение текста примера 17_1 в браузере.

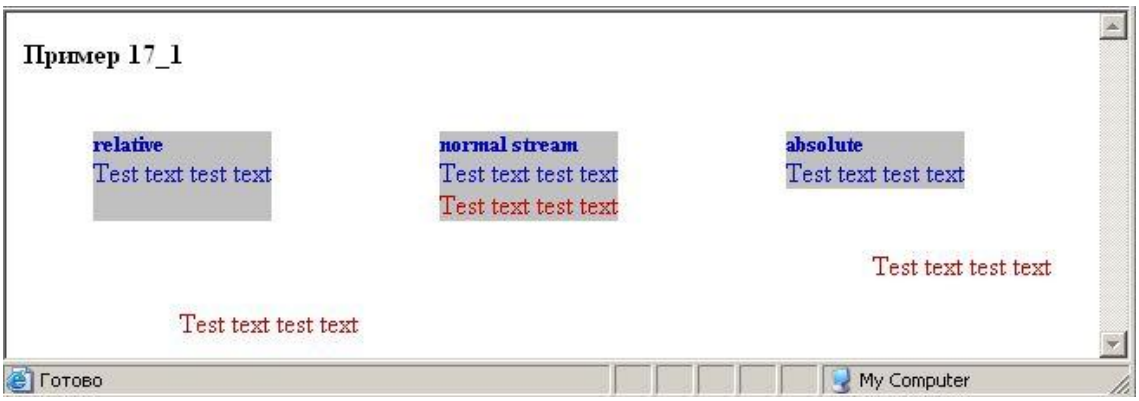


Рис.3.16

В случае относительного позиционирования текст красного цвета отстоит на **top:70 и left:50** от своего несмещенного положения в потоке элементов, а в случае абсолютного – от родительского элемента, которым является слой **div**.

В примере 17_2 показано использование **абсолютного** позиционирования для отображения двух слоев. Вертикальный слой цвета **deeppink** содержит логотип некоторого ЗАО, который позиционирован относительно его нормального положения в потоке элементов страницы, а также текст "ЗАО Воздушный шар", имеющий оформление, описанное в классе "**revers**".

Горизонтальный слой цвета **mistyrose** расположен поверх вертикального и содержит текст, оформленный согласно стилю **bluetext**.

Пример 17_2

```
<html><head><title>Пример 17_2</title>
<style type="text/css">
.revers {
font-weight:bold;
color:white;
text-align: center;}
.bluetext {
font-weight:bold;
color:darkblue;
text-align: justify;
padding:0.5em;
margin:0}
</style>
</head>

<body>
<strong>Пример 17_2</strong>
<!-вертикальныйслой-->
<div style="position: absolute; top:0;
left: 250;width:200;
height:300;background:'deeppink'">

<img src="ball.jpg" style="position:relative; top:20;
left:25;" >
<p class="revers">ЗАО Воздушный шар</p>
</div>
<!-горизонтальный слой-->
<div style="position: absolute; top:155;
left:25;width:400;
```

height:90;background :'mistyrose' ">

<pclass="bluetext">Слой цвета mistyrose с абзацем лежит поверх правой колонки (слоя цвета deeppink) с логотипом. Изображение логотипа позиционировано относительно (relative) его нормального положения в потоке элементов страницы.</p>

** **

</body>

</html>

На рис.3.17 показано отображение текста примера 17_2 в браузере.

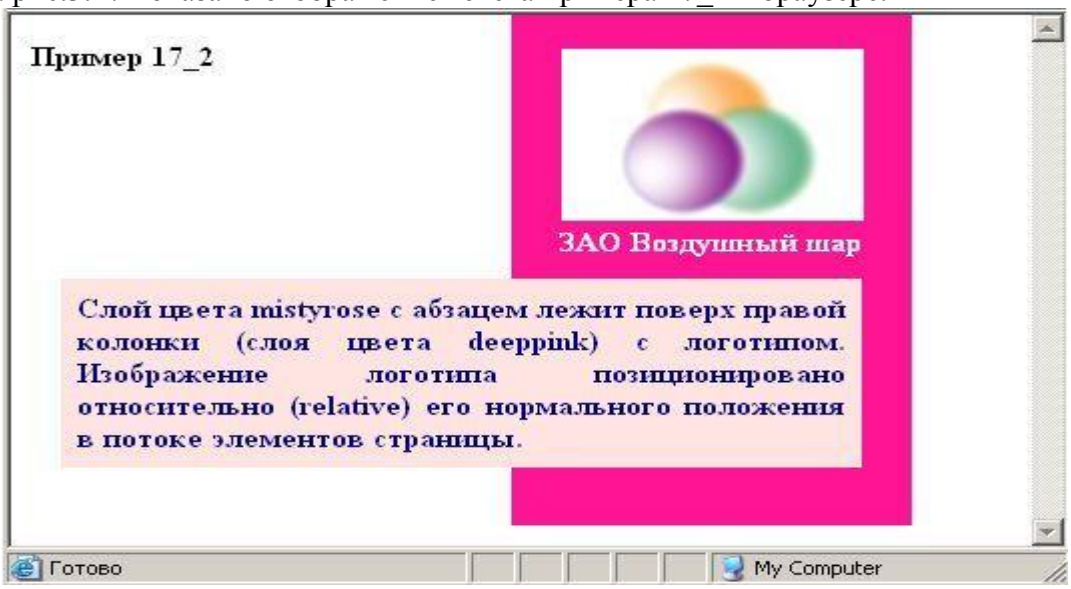


Рис.3.17

Пример 17_3 демонстрирует использование свойства **z-index**, которое позволяет задать порядок (уровень) расположения элемента. Элемент ** **используется дляопределения свойств слоя. Он является контейнерным элементом и позволяет применить позиционирование и **z-index** для одного или нескольких элементов, содержащихся в нем.

В примере 17_3 на **Web**-странице выводится текст и слой ** **серого цвета**.**

Пример 17_3

```
<html><head><title>Пример 17_3</title>
<style type="text/css">
<!--
.div_layer  {
background-color:#c0c0c0;
position:absolute;left:80;
width:150; height:100;
visibility:visible;}
.div_text {
color:blue;
position:absolute;left:50;
width:350; height:100;
font-size:24pt;}
-->
</style>
</head>

<body>
<strong>Пример17_3</strong>
```

** **
** **
** z-индекс слоя больше, чем текста. Слой расположен "ближе" к пользователю. **

<divclass="div_layer" style="top:200;z-index:-1;">
** **
** z-индекс текста больше,чем слоя. Текст расположен**
**"ближе" к пользователю. **
</body></html>

На рис.3.18 показано отображение текста примера 17_3 в браузере.

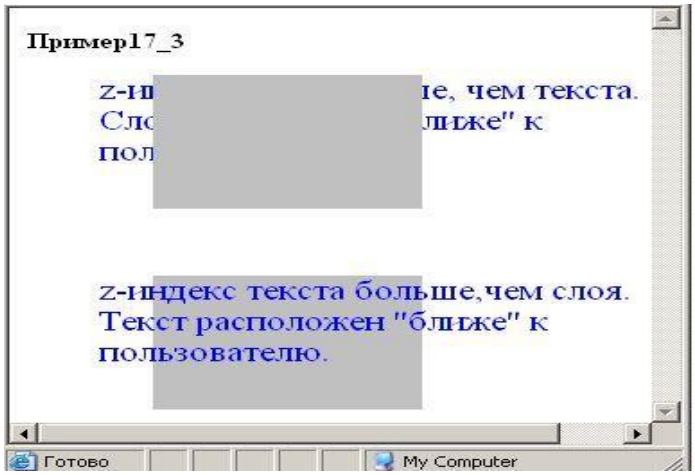


Рис.3.18

Свойства слоя серого цвета и слоя текста описаны с помощью соответствующих классов в разделе описания стилей (<**style**>). В верхней части примера **z-index** серого слоя положителен (равен 1), тогда как текст имеет по умолчанию **z-index** равный 0. В результате слой текста находится на заднем плане, а серый слой – на переднем плане.

В нижней части примера **z-index** серого слоя отрицателен (равен –1), а текст лежит в нулевом слое. При этом текст оказывается на переднем плане, а серый слой – за текстом.

В примере 18 демонстрируется эффект, благодаря которому можно управлять представлением информации в пределах ограниченной области. Ограниченная область меньше, чем изображение, а переполнение скрыто. Таким образом, изображение вписывается в ограниченную область. Область с переполнением помещена ниже обычного изображения, которое окружено тонкой рамкой, а область с переполнением – рамкой с большей толщиной.

Пример 18
<html>
<html><head><title>Пример 18</title>
<style type="text/css">
<!--
.overflow {
position: absolute; top:190;left: 50;
width:80;
height:80;
border:2px solid black;
overflow:hidden;

```
}
-->
</style>
</head>

<body>
<strong>Пример18</strong><br>
<img  src="flower.gif" width=150 height=150 alt=”flower” border=1>
<img class="overflow" src="flower.gif" width=150 height=150 alt=”flower”>
</body>
</html>
```

На рис.3.19 показано отображение текста примера 18 в браузере.

В примере 19 использован класс **clip1**, который описывает текст, помещенный в квадратную область, описанную дескриптором ** .** Класс **clip** обеспечивает применение к этой области отсекания (сверху и слева на 25 пикселей, а снизу и справа на 125 пикселей). На рис.3.20 текст в квадратной области расположен справа, а отсеченный – слева.

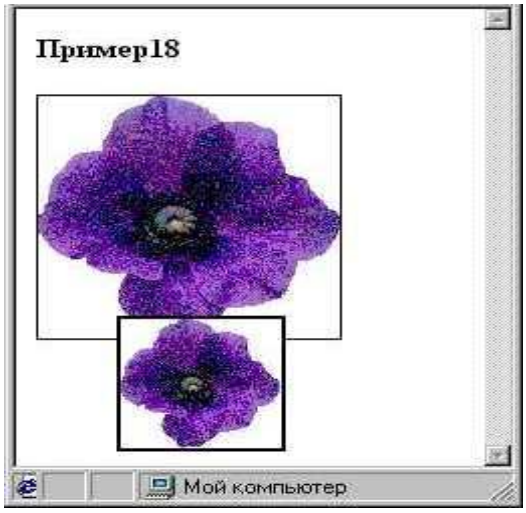


Рис.3.19

Пример 19

```
<html><head><title>Пример 19</title>
<style type="text/css">
<!--
.clip {
position: absolute; top:50;
left: 230;
width:150;
height:150;
color:yellow;
background-color:black;
clip:rect(25px 125px 125px 25px);}
.clip1{
position: absolute; top:50;
left: 30;
width:150;
height:150;
```

```
color:yellow;
background-color:black;
}
-->
</style>
</head>

<body>
<strong>Пример19</strong>
<div class="clip1">Текста абзаца будет обрезан при помощи отсекания, примененного к данной квадратной области размером 150 на 150 пикселей.</div>
<div class="clip">Текста абзаца будет обрезан при помощи отсекания, примененного к данной квадратной области размером 150 на 150 пикселей.</div>
</body></html>
```

На рис.3.20 показано отображение текста примера 19 в браузере.

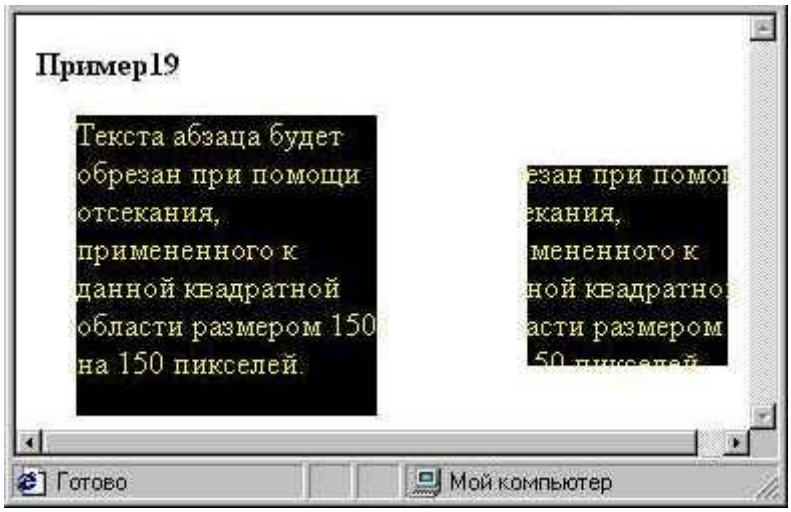


Рис.3.20

В примере 19_1 показано применение свойства **display** для создания горизонтального меню из списка ссылок. *Блочный* элемент, которым является список ссылок, преобразован в *строчный* с оформлением, соответствующим элементам меню.

```
Пример 19_1
<html><head><title>Пример 19_1</title>
<style>
#nav h5 {text-align:center;font-size:16pt}
#nav ul {padding: 3px 0;
border-bottom:1px solid #778;
font :bold 12px Verdana, sans-serif;
}

#nav ul li {
display: inline;
}
```

```
#nav ul li a {
padding: 3px 0.5em;
margin-left: 3px;
border: 1px solid #778;
border-bottom: none;
background: #dde;
text-decoration: none;
}
#nav ul li a:link {
color: #0000ff;
}

#nav ul li a:visited {
color: #667;
}
 #nav ul li a:link:hover,  #nav ul li a:visited:hover {
color: #000;
background: #aae;
border-color: #227;}

#nav ul li a#current {
background: white;
border-bottom: 1px solid white;
}
</style>
</head>
<body>
<div id="nav">
<h5> navigation bar <h5>
<ul>
<li><a href="bg1.html">home</a></li>
<li><a href="bg1.html">about</a></li>
<li><a href="bg1.html">archives</a></li>
<li><a href="bg1.html"  id="current">contact</a></li>
</ul>
</div>
```

</body></html>

На рис.3.21 показано отображение текста примера 19_1 в браузере.

Рис.3.21

3.10. Фильтры

Начиная с выхода версии **InternetExplorer 4,** фирма **Microsoft** предложила разработчикам **Web**-страниц два типа специальных эффектов для изменения вида текста и графики, которые называются **VisualFilter** (визуальные или статические фильтры) и **TransitionsFilter** (динамические фильтры).

Эффекты, предлагаемые фильтрами, разнообразны – от зеркального отражения текста и графики до исчезновения их наслучайным образом формирующейся шахматной доске.

ЛАБОРАТОРНО РАБОТА №1

Задание 1_HTML

Цель: Изучение возможностей структурирования и форматирования текста.

1. Создайте шаблон, пользуясь программой "**HomeSite**" и скорректируйте его по приведенному образцу и сохраните его на своей дискете под именем **шаблон.html.**

```
<html>
<head>
        <title>…</title>
<script language="JavaScript" type="text/javascript">
<!- -
//-->
</script>
<style type="text/css">
<!- -
-->
</style>
</head>
<body>
</body>
</html>
```

2. Откройте файл **шаблон.html**и сохраните его под именем **Имя_1.html** в папке, созданной для **HTML** файлов на своей дискете, где Имя – ваше имя или фамилия.

3. Введите *заглавие* и два *заголовка* первого и второго уровней, пользуясь дескрипторами **<title> и <h1> и <h2>.**

4. Введите *абзац* текста, состоящий из 2-3-х строк и скопируйте его трижды.

5. Для форматирования текста:

5.1. Установите цвет текста **(#663333**) и фона **(#FFFFCC**) или другие цвета из безопасной палитры в дескрипторе **<body>.**

5.2. В первом абзаце с помощью дескрипторов *физического* форматирования **<i>,<b>, <small>, <big>** отформатируйте фрагменты абзаца курсивом, полужирным начертанием, а также уменьшенным и увеличенным кеглем. Установите для абзаца шрифт **Courier** размером 4, пользуясь дескриптором <font> и просмотрите результат в браузере.

5.3. Во втором абзаце используйте дескрипторы *физического* форматирования **<sup>, <sub>, <u>** для форматирования фрагментов абзаца как верхнего и нижнего индексов и подчеркивания. Установите для абзаца шрифт **Verdana** размером 3 произвольного цвета и просмотрите результат в браузере.

5.4. В третьем абзаце используйте дескрипторы *логического* форматирования **, и <ins>** для форматирования фрагментов курсивом, полужирным начертанием и подчеркиванием. Установите для абзаца шрифт **Garamond** размером 3 произвольного цвета и просмотрите результат в браузере.

5.5. В четвертом абзаце выделите фрагмент дескрипторами <**pre**> и отформатируйте текст произвольным образом, вставив в него две пустые строки. Просмотрите результат в браузере.

5.6. Добавьте строку текста, содержащую сведения о студенте, выполняющем данное задание (ФИО, адрес, E-mail) и разделите текст на три строки, пользуясь дескриптором **
.** Просмотрите результат в браузере.

6. Используйте средства выравнивания текста.

6.1. Первый абзац выровняйте по ширине, пользуясь атрибутом **align="justify**" и установив для абзаца закрывающий дескриптор **</p>.** Просмотрите результат в браузере.

6.2. Второй абзац заключите в контейнерный дескриптор <**blockquote**>. Просмотрите результат в браузере.

6.3. Третий и четвертый абзацы заключите в контейнерный дескриптор ** ** и выровняйте по центру. Просмотрите результат в браузере.

Задание 2_HTML

Цель: Изучение средств связывания страниц.

Часть 1. Средства связывания страниц в **HTML**.

1. Научиться формировать пути в относительных**URL**.

1.1 Создайте два файла **index.html и image.gif.** Файл **index.html** должен содержать гиперссылку на файл **image.gif**, которая видоизменяется в зависимости от расположения файлов. Три варианта расположения файлов представлены на рисунках 2. (а, б, в).

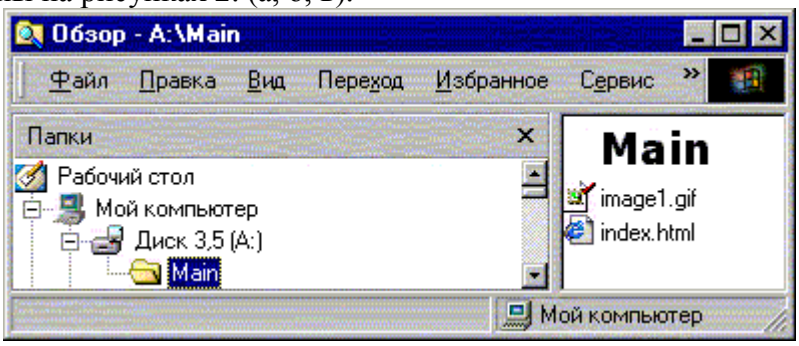


Рис.2.а

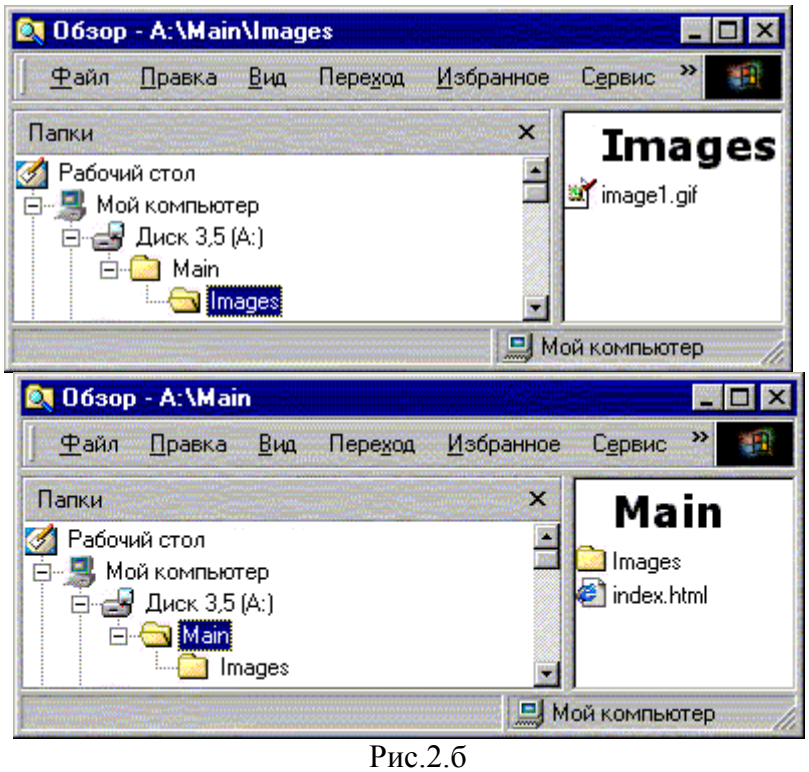


Рис.2.б

1.2 Проверьте работоспособность гиперссылок.

2. Научитесь пользоваться внутренними ссылками.

2.1 В файле **index.html** введите заголовок **"Глава 1"** и несколько абзацев текста (пользуясь копированием увеличьте документ до двух – трех экранов).

2.2 В конце документа установите ссылку

<ahref="#Гл1">В начало главы

2.3 Заголовок **"Глава 1"** сделайте *именованным элементом привязки* (*якорем*)

<aname="Гл1"><h1>Глава1</h1>

2.4 Проверьте работоспособность ссылки.

2.5 Создайте в графическом редакторе или позаимствуйте в Интернете изображение стрелки, ведущей вверх (например, ) и используйте изображение как изображение – ссылку. Расположите изображение стрелки дважды: внизу страницы и в середине текста.

В этом случае ссылка будет иметь вид:

<img src="uparrow.gif" alt="Вернуться в начало" width=20 height=25>

2.6 Проверьте работоспособность ссылок.

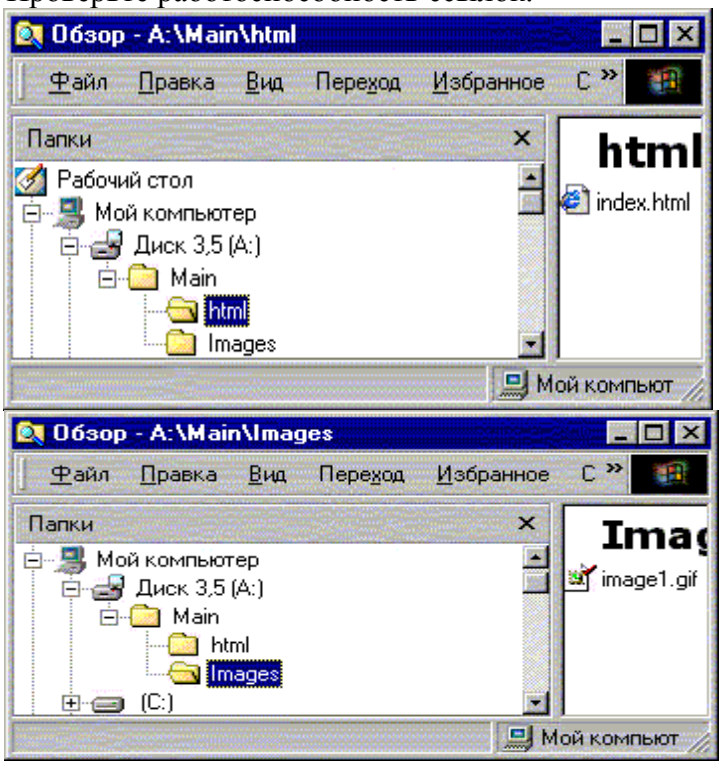


Рис.2.в

3. Научитесь использовать почтовую ссылку.

3.1.Создайте в конце документа **index.html** ссылку на свой почтовый адрес и проверьте ее работоспособность. Ссылка должна иметь вид

<ahref=mailto: user@server>Написать письмо touser

4. Предъявите работу преподавателю.

Часть 2. Использование **CSS** для оформления гиперссылок.

1. Создайте в папке **Main** четыре файла **index.html, 1.html, 2.html и 3.html.**

2. В файле **index.html**создайте список ссылок на **1.html, 2.html и 3.html.**

3. При наведении мыши на каждую ссылку должна появляться подсказка, содержащая информацию о том файле, на который производится ссылка.

Подсказка формируется с помощью атрибута **title** соответствующего дескриптора.

4. Объявите стиль ссылок, установив *цвет* и *отсутствие подчеркивания*, объявление стиля выглядят следующим образом: **a {color: …;text-decoration: none}**

5. Пользуясь псевдоклассом, установите *цвет, увеличенный кегль* и *подчеркивание* для ссылок при *наведении* курсора, объявление выглядит следующим образом:
a :hover {color: …; font-size:…; text-decoration:underline}

6. Проверьте работу ссылок и предъявите задание преподавателю.

Задание 3_HTML

Цель: использование таблиц для структурирования информации и создания макетов страницы.

1. Создайте **html** документы, позволяющие построить таблицы, приведенные на рисунках 2.3-2.6.

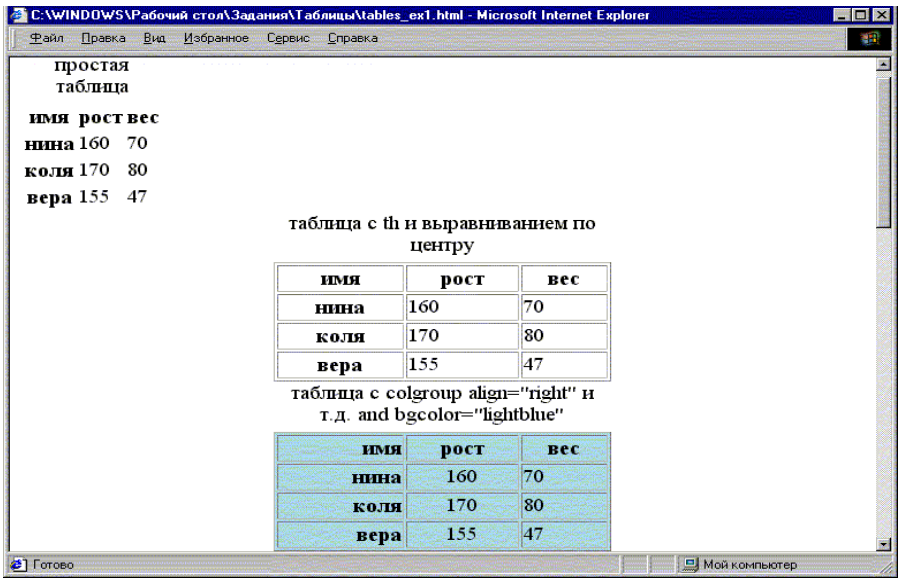


Рис.2.3

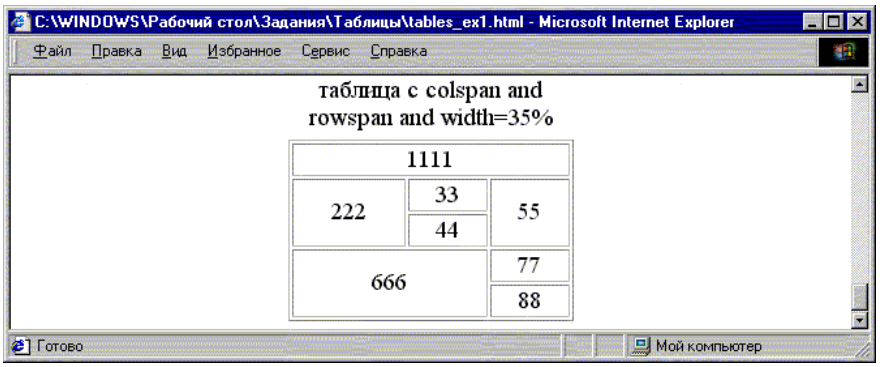


Рис.2.4

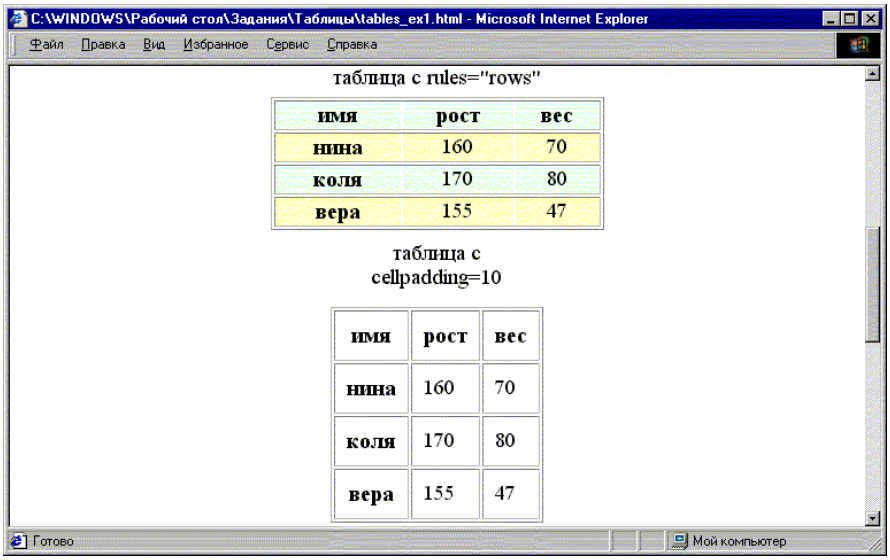


Рис.2.5.

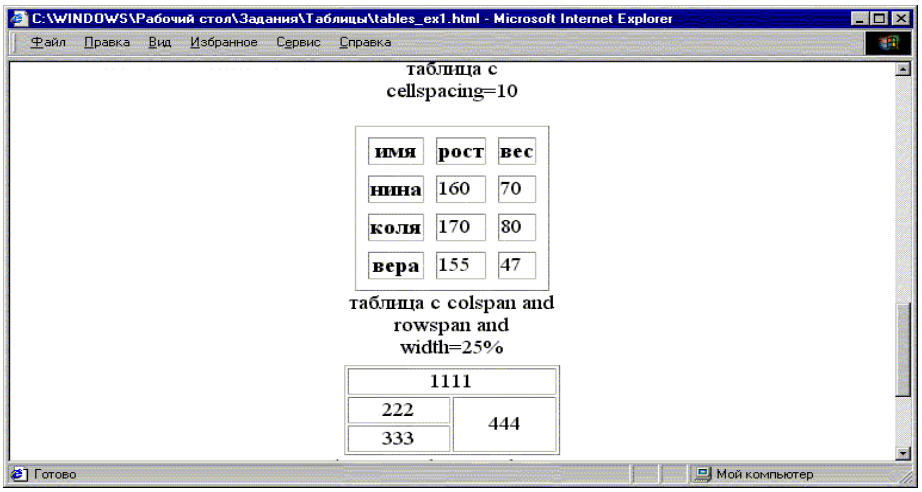


Рис.2.6.

2 Создайте **HTML** документ, в котором расположена таблица, содержащая две строки, в первой находится заголовок – "Расписание занятий", а во второй – названия дней недели от понедельника до субботы. Используйте названия дней недели как гиперссылки на соответствующий фрагмент расписания, созданный в отдельном файле.

3 Создайте **HTML** документ, в котором таблица используется для макетирования страницы так, как показано на рис.2.7 и 2.8.

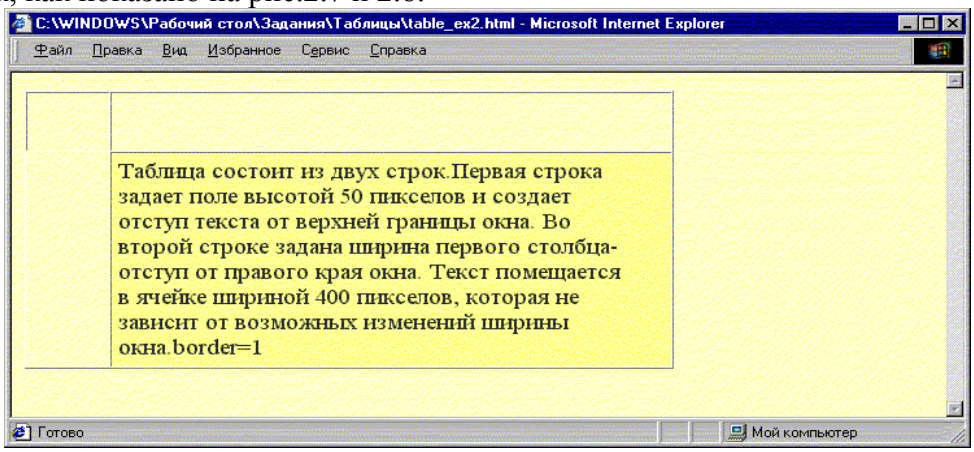


Рис.2.7

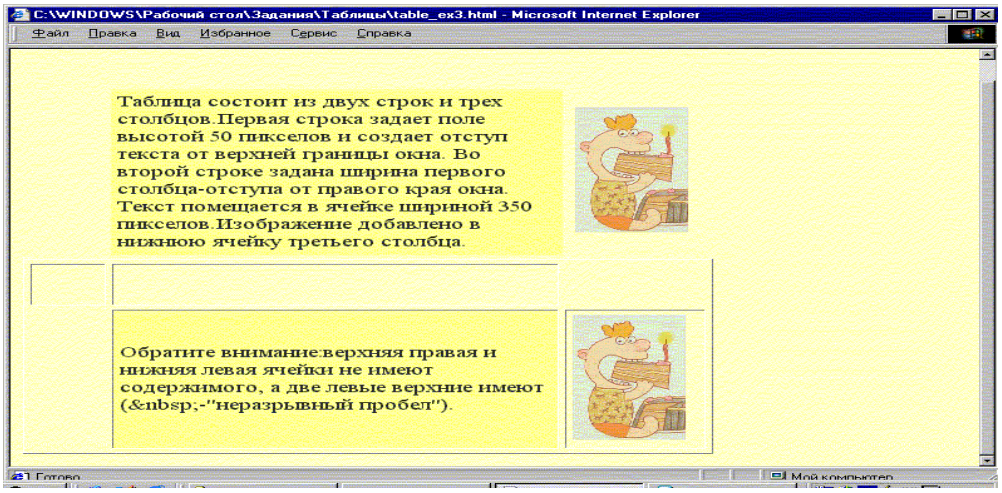


Рис.2.8

4. Создайте **HTML** документ, позволяющий построить таблицу, приведенную на рис.2.9.

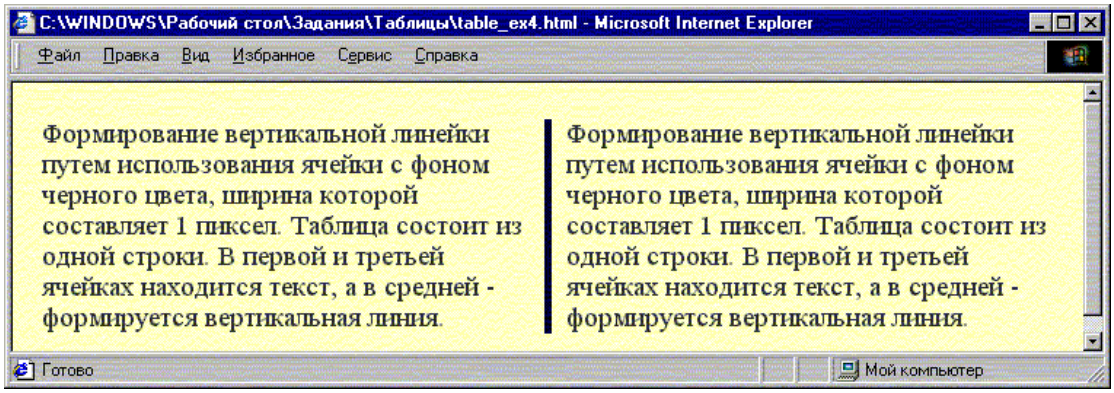


Рис.2.9

ЛАБОРАТОРНО РАБОТА №2

Задание 1_HTML

Цель: изучение способов выравнивания графических изображений на странице.

1. Подготовьте небольшое графическое изображение 50×50 px или воспользуйтесь файлом **ducky.gif**.
2. Создайте **HTML** документ **align_имя_1.html**, позволяющий поместить в абзац текста изображение **ducky.gif**

<p>Данный абзац текста содержит маленькую
<imgsrc= ducky.gifalt=duckywidth=50 height=50>
графическую вставку внутри текста.

3. Просмотрите результат в браузере.
4. Трижды скопируйте данный абзац и используйте три типа выравнивания изображения, *находящегося в тексте*, относительно базовой линии строки в каждом абзаце: **align="bottom"; align="top"; align="middle".**
5. Просмотрите результат в браузере.
6. Используйте выравнивание графического изображения, расположенного *вне* текста. Для этого создайте конструкцию

** <imgsrc= ducky.gifalt=duckywidth=50 height=50 align="right">**В рассматриваемом примере изображение выравнивается по правому краю страницы и находится справа от текста. Текст при этом обтекает изображение, так как это имело место в текстовом процессоре Word, когда изображение не находилось в тексте, а было расположено в своем слое.** **

7. Просмотрите результат в браузере.

8. Скопируйте фрагмент **html** документа и примените выравнивание по левому краю. Измените соответственно текст.

9. Примените атрибуты установки промежутков между текстом и изображением. Поместите изображение внутрь текста абзаца и увеличьте текст. Затем примените атрибуты изображения **vspace=10 и hspace=20.**

10. Просмотрите результат в браузере.

11. Используйте разделитель строк **
**

11.1. Создайте следующие фрагменты **html** текста и объедините в документ **align_имя_2.html**

** <imgsrc= ducky.gifalt=duckywidth=50 height=50>**
первая строка
**
вторая строка**
**
третья строка**

11.2. Просмотрите результат в браузере. Обратите внимание на то, что изображение является частью текста.

11.3.Добавьте фрагмент

** <imgsrc= ducky.gifalt=duckywidth=50 height=50 align="left">**
первая строка
**
вторая строка**
**
третья строка**

11.4. Просмотрите результат в браузере.

11.5. Добавьте фрагмент, в котором к дескриптору**
добавляется атрибутclear="left",** позволяющий прервать текст и расположить его ниже изображения

** <imgsrc= ducky.gifalt=duckywidth=50 height=50 align="left">**
первая строка
**
вторая строка**
<brclear="left">третья строка

11.6. Просмотрите результат в браузере.

11.7.Добавьте фрагмент с атрибутом дескриптора**
clear="right" и** атрибутами**align="right"** дескрипторов ** и <img>**

11.8. Просмотрите результат в браузере.

11.9. Добавьте фрагмент, содержащий конструкцию**clear="all"** следующего содержания:

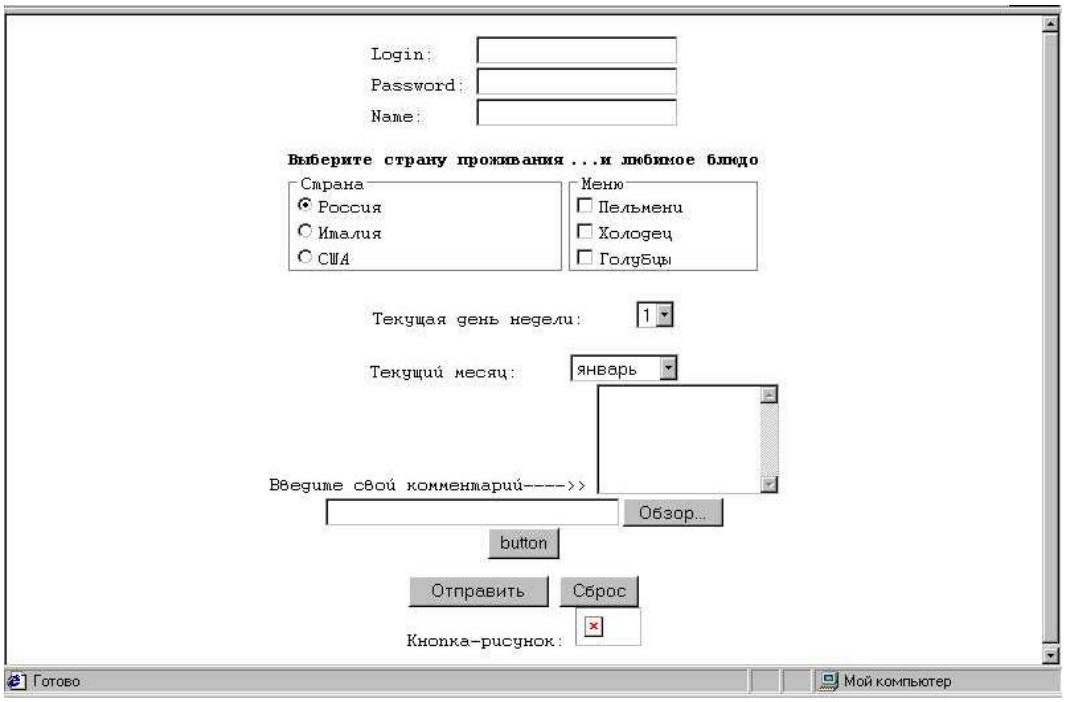
<palign="center">
<imgsrc= ducky.gifalt=duckywidth=50 height=50 align="right">
<imgsrc= ducky.gifalt=duckywidth=50 height=50 align="left">
первая строка
**
вторая строка**
<brclear="all">третья строка

11.10. Просмотрите результат в браузере

Задание 2_HTML

Цель: Знакомство с построением форм.

1. Создайте форму, приведенную на рис.2.10, пользуясь методическими и справочными материалами и предъявите созданный файл преподавателю.



ЛАБОРАТОРНО РАБОТА №3

Задание 1_CSS

Цель: Знакомство со способами применения стилей в документе HTML. Использование классов и группирования.

1. Познакомьтесь с возможностью *встраивания* стиля в документ **HTML**Создайте **HTML**-документ, используя в качестве содержимого текст примеров 1-3. Просмотрите полученные результаты в браузере.

2. Познакомьтесь с возможностью *внедрения* стиля в документ **HTML**Создайте **HTML**-документ, используя в качестве содержимого текст примера 4. Просмотрите полученные результаты в браузере.

3. Познакомьтесь с использованием *классов* в документе **HTML** Создайте **HTML**-документ, используя в качестве содержимого текст примеров 5 и 6. Просмотрите полученные результаты в браузере.

4. Создайте в том же документе два новых класса для оформления заголовка второго уровня и произвольного фрагмента текста. Примените созданные классы и просмотрите результаты в браузере.

5. Познакомьтесь с использованием *селектора* **ID** в объявлении стиля в документе **HTML** создайте **HTML**-документ, используя в качестве содержимого текст примера 8. Измените объявление стиля. Просмотрите полученные результаты в браузере.

ЛАБОРАТОРНО РАБОТА №4

Задание 1_CSS

Цель: Знакомство со свойствами позиционирования и визуализации **CSS**. Использование статических и динамических фильтров.

1. Познакомьтесь с использованием *позиционирования* и *визуализации* **CSS**.

1.1 Создайте **HTML**-документ, используя в качестве содержимого текст примера 17. Просмотрите полученные результаты в браузере.

1.2 Измените значение **z-индекса** в примере 17 так, чтобы *верхний* текст оказался впереди соответствующего слоя. Просмотрите полученные результаты.

1.3. Установите значение свойства **visibility** текстового слоя так, чтобы текст стал невидимым.

1.4. Измените свойства слоя и текста так, чтобы текст позиционировался, начиная с позиции **top**:50 **left**:50, а серый слой позиционировался относительно текста на расстоянии 100 пикселов левее текста.

1.5. Создайте **html**-документ, используя позиционирование и **z-индекс** для воспроизведения заголовка , приведенного на рис.3.18.

1.6. Создайте **HTML**-документ, используя в качестве содержимого текст примера

Рис.3.18.

1.7. Измените позиционирование вертикального слоя в примере 16 и расположите его в левой части окна браузера.

1.8. Измените позиционирование логотипа произвольным образом.

2. Познакомьтесь со свойствами переполнения (**overflow**) и отсекания (**clip**). Создайте **HTML**-документы, используя в качестве содержимого текст примеров 18 и 19. Просмотрите полученные результаты в браузере. Измените область отсекания в примере 19. Просмотрите полученные результаты в браузере.

3. Познакомьтесь с использованием статических фильтров. Создайте **HTML**-документ, используя в качестве содержимого текст примера 20. Просмотрите полученные результаты в браузере. Измените тип применяемых фильтров на **blur** и **flipv**. Познакомьтесь с остальными фильтрами, получив их параметры из справочной таблицы учебного пособия. Просмотрите полученные результаты в браузере.

4. Познакомьтесь с использованием динамических фильтров. Создайте **HTML**-документы, используя в качестве содержимого тексты примеров 21.1 и 21.2. Просмотрите полученные результаты в браузере. Измените тип применяемых фильтров, заменив значение параметра **Transition** на число из диапазона от 0 до 23. Эти параметры можно получить из справочной таблицы учебного пособия. Просмотрите полученные результаты в браузере. Предъявите результаты работы преподавателю.

ЛИТЕРАТУРА

1. Васильев, В. В., Сороколетова, Н. В. Практикум по Web-технологиям: практикум для вузов М.: ФОРУМ, 2013 Смирнов С.Н. Электронный бизнес.−-М.:ДМК Пресс, 2003.
2. Грабауров В.А. Электронный бизнес. − БГЭУ, 2007
3. Дунаев В. HTML, скрипты и стили. − СПб: BHV, 2005
4. Кобелев О.А. Электронная коммерция. – Издательский дом «Дашков и К». 2008.
5. Кирсанов Д. Веб-дизайн: книга Дмитрия Кирсанова. – СПб: Символ-Плюс, 2001.
6. Круг С. Веб-дизайн: книга Стива Круга или «не заставляйте меня думать!», 2-е издание.-Пер. с англ.−СПб: Символ-Плюс, 2008.
7. Муссиано Ч., Кеннеди Б. HTML и XHTML. Подробное руководство, 6-е издание. Пер. с англ .− СПб: Символ-Плюс, 2008
8. Мархвида И.В. Создание Web-страниц: HTML, CSS, JavaScript. – Мн.:Новое знание, 2002.
9. Мейер Э. CSS-каскадные таблицы стилей. Подробное руководство, 3-е издание.-Пер. с англ.−СПб:Символ-Плюс, 2008.
10. Нильсен Я. Веб-дизайн: книга Якоба Нильсена. – СПб: Символ-Плюс, 2001.
11. Стригина Е.В. Web-дизайн. Учебное пособие. . − СПб: СПбГУТ, 2004
12. Советов, Б. Я., Цехановский, В. В., Санкт-Петербург. гос. электротехн. ун- т "ЛЭТИ" Информационные технологии: учебник для бакалавров М.: Юрайт, 2012
13. Хольцшлаг Молли. Использование HTML 4, 6-е издание. Специальное издание. Пер. с англ. :Уч. пос. - М.: Издательский дом "Вильямс", 2001.
14. Чебыкин Р. Самоучитель HTML и CSS. Современые технологии. − СПб: BHV, 2008

Интернет ресурсы:

1. Сайт по HTML и CSS – Режим доступа: http://htmlbook.ru/
2. Портал W3Schools. – Режим доступа: http://w3schools.com/