

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

Воронежский государственный лесотехнический университет имени Г.Ф. Морозова

Лавлинский В.В., Табаков Ю.Г., Курипта О.В.

**Web-инжиниринг**

Часть 1. Основы HTML технологий

Лабораторный практикум

для направления 230200.62 Информационные системы и 230.400.62 Информационные системы и технологии

Воронеж 2015

Оглавление

[**Предисловие** 4](#_Toc427751304)

[**Введение в язык HTML** 6](#_Toc427751305)

[**Лабораторная работа №1. Текстовое оформление страниц** 11](#_Toc427751306)

[**Задание к выполнению лабораторной работы** 13](#_Toc427751307)

[**Контрольные вопросы** 14](#_Toc427751308)

[**Лабораторная работа №2. Создание гиперссылок** 15](#_Toc427751309)

[**Задание к выполнению лабораторной работы** 17](#_Toc427751310)

[**Контрольные вопросы** 20](#_Toc427751311)

[**Лабораторная работа №3. Создание списков** 21](#_Toc427751312)

[**Задание к выполнению лабораторной работы** 23](#_Toc427751313)

[**Контрольные вопросы** 25](#_Toc427751314)

[**Лабораторная работа №4. Вставка изображений в html-страницы** 27](#_Toc427751315)

[**Задание к выполнению лабораторной работы** 28](#_Toc427751316)

[**Контрольные вопросы** 29](#_Toc427751317)

[**Лабораторная работа №5. Создание таблиц** 30](#_Toc427751318)

[**Задание к выполнению лабораторной работы** 31](#_Toc427751319)

[**Контрольные вопросы** 36](#_Toc427751320)

[**Лабораторная работа №6. Создание форм** 37](#_Toc427751321)

[**Задание к выполнению лабораторной работы** 43](#_Toc427751322)

[**Контрольные вопросы** 43](#_Toc427751323)

[**Лабораторная работа №7. Создание фреймов** 44](#_Toc427751324)

[**Задание к выполнению лабораторной работы** 45](#_Toc427751325)

[**Контрольные вопросы** 47](#_Toc427751326)

[**Лабораторная работа №8. Каскадные таблицы стилей CSS** 49](#_Toc427751327)

[**Задание к выполнению лабораторной работы** 52](#_Toc427751328)

[**Контрольные вопросы** 54](#_Toc427751329)

[**Литература** 55](#_Toc427751330)

**Предисловие**

В последние два десятилетия Интернет прочно вошел в нашу жизнь. В наше время он охватывает почти все сферы жизни: культура, искусство, бизнес. Наряду с обычными музеями, библиотеками, магазинами все большую популярность во всем мире завоевывают аналогичные Интернет-ресурсы. Даже бизнес становится виртуальным: создаются Web-представительства фирм, электронные магазины и торговые площадки, электронные банки, которые дают возможность осуществлять типичные для этих сфер операции с большей скоростью. В связи с этим профессия Web-дизайнера и Web-программиста стала востребованной и у нас в России. Интернет-технологии создания web-ресурсов требует владения такими языками для написания серверных скриптов, как Perl, PHP, ASP, владения языком XML, а также умения создавать динамические страницы и сайты, используя скрипты VBScript и JavaScript, которые позволяют осуществлять вычисления, работу с датой и временем, изменение элементов страницы по желанию пользователя.

Perl – язык для написания CGI-скриптов (Common GateWay Interface). CGI-скрипит – это программа, которая выполняется на Web-сервере по запросу посетителя сайта. CGI – это не язык программирования, а специальный интерфейс, с помощью которого и происходит запуск скрипта и взаимодействие с ним.

PHP-скрипт (Hypertext Preprocessor) – скрипт, написанный на языке PHP и внедренный в HTML-документ.

ASP-скрипт (Active Server Pages) – скрипт, написанный на языке Visual Basic Scripting Edition и внедренный в HTML-документ.

На первом этапе необходимо научиться создавать простые HTML-страницы. Не стоит считать, что в этом нет необходимости. Простого HTML вполне хватит для двухстраничной коммерции, которая распространяется сейчас по всему миру. HTML позволяет создать «заготовки» страниц, а уже потом можно добавлять в них различные скрипты. XML же – это надстройка над HTML. Поэтому без изучения HTML не обойтись. Итак, HTML – Hyper-Text MarkUp Language – язык разметки гипертекста. Гипертекст – расширенный текст, который может содержать в себе текст, иллюстрации, различные внедренные объекты и ссылки на другие ресурсы или на другие Web-страницы.

При передаче информации через Интернет используются как раз HTML-страницы – файлы \*.htm и \*.html, поскольку они содержат в себе гипертекст, имеют размер меньше, чем текстовые или иные файлы и для их просмотра нужен только браузер.

HTML-документ представляет собой обычный текстовый документ с управляющими конструкциями языка HTML – тегами, которые и производят действия « форматирования» над текстовыми блоками и осуществляют вставку различных объектов в документ.

Создавать HTML-страницы можно вручную путем самостоятельного написания тегов в текстовом редакторе или используя Web-редакторы с автоматизацией ввода нужных тегов. Например, WebEdit, Arachnophilia или же используя программы, которые создают теги сами в ответ на действие пользователя на странице, по принципу What You See Is What You Get, например, MS Word, MS FrontPage. Но они задают абсолютное форматирование и часто дописывают много лишнего кода к странице. К тому же для создания более уникальных страниц ими не обойдешься, для этого лучше писать теги HTML самому и использовать каскадные таблицы стилей.

Данный практикум содержит в себе все основные разделы, необходимые для создания Web-страниц. В каждом разделе есть примеры использования тегов, показан результат действия этих тегов, а также представлены лабораторные работы. Практикум можно использовать для самостоятельного изучения технологии создания статических Web-страниц студентам любых специальностей или тем, кто хочет самостоятельно изучить основы языка HTML.

**Введение в язык HTML**

Автономные Web-документы используют язык HTML (Hypertext Markup Language – язык разметки гипертекста). Гипертекст, то есть расширенный текст, включает дополнительные элементы: иллюстрации, ссылки, вставные объекты. Под разметкой понимается использование специальных кодов, легко отделяемых от смыслового содержания документа и используемых для реализации гипертекста. Применение этих кодов подчиняется строгим правилам, определяемым спецификацией языка HTML.

HTML-документ – это файл, содержащий обыкновенный текст со специальными командами. Такой файл может быть подготовлен в произвольном текстовом редакторе (существуют, однако, специальные программы-конверторы и HTML-редакторы).

HTML-документ состоит из содержимого, то есть собственно полезной информации, и команд, задающих структуру.

Каждая команда (управляющая конструкция) HTML-документа (тег) должна заключаться в угловые скобки, вот так: <тег>. Чаще всего в документе встречаются парные теги (открывающий и соответствующий ему закрывающий), так как браузеру необходимо знать область действия тега. Существуют и одиночные теги, однако, используются они только там, где область действия очевидна и дополнительной информации не требуется (ясно, например, что если мы встретили тег «начало абзаца» (**<**Р**>**), то предыдущий абзац уже закончился). В сомнительном же случае лучше перестраховаться и поставить закрывающий парный тег, иначе документ может оказаться нечитаемым. Открывающий и закрывающий теги называются одинаково и отличаются друг от друга только символом «наклонная черта» или «слэш» – «/», который ставится сразу после открывающей угловой скобки закрывающего тега. Закрытие парных тегов выполняется так, чтобы соблюдались правила вложения.

<В><I>На этот текст воздействуют два тега</I></B>

Кроме того, тег может включать атрибут, дающий дополнительную информацию браузеру. Например, при помощи атрибута можно попросить браузер изменить величину шрифта, ориентацию изображения по отношению к строке следующего за ним текста, поменять цвет фона документа и т.д. В парных тегах атрибуты добавляются только к открывающему тегу. Атрибуты представляют собой дополнительные ключевые слова, отделяемые от ключевого слова, определяющего тег, и от других атрибутов пробелами и размещаемые до завершающего тег символа «>». Способ применения некоторых атрибутов требует указания значения атрибута. Значение атрибута отделяется от ключевого слова атрибута символом «=» (знак равенства) и заключается в кавычки.

<Н1 ALIGN="LEFT">

Язык HTML, в большинстве случаев, совершено равнодушен к регистру, в котором набираются теги. Скажем, браузеру совершенно все равно, наберете вы тег, служащий для рисования горизонтальной линии, как **<**HR**>** или **<**hr**>**, эффект будет один и тот же.

HTML не признает никакого дополнительного форматирования текста, кроме как с помощью тегов. В результате текст, превосходно смотрящийся в текстовом редакторе, в окне браузера сольется в единую нечитаемую массу. Так, на месте нескольких пробелов будет лишь один пробел. Исчезнут все заголовки, пустые строки, деление текста на абзацы. Без HTML-тегов браузер просто игнорирует все элементы форматирования.

Определение HTML как языка разметки основывается на том, что при удалении из документа всех тегов получается текстовый документ, совершенно эквивалентный по содержанию исходному гипертекстовому документу. Таким образом, при отображении документа HTML сами теги не отображаются, но влияют на способ отображения остальной части документа.

**Специальные символы**

В таблице приведены некоторые специальные символы HTML, которые имеют особое назначение и собственный способ представления в виде мнемонического или числового кода.

|  |  |  |  |
| --- | --- | --- | --- |
| **Символ** | **Мнемокод** | **Числовой код** | **Описание** |
|  | &nbsp; | &#160; | неразрывный пробел |
| ¢ | &cent; | &#162; | цент |
| £ | &pound; | &#163; | фунт стерлингов |
| ¥ | &yen; | &#165; | иена или юань |
| § | &sect; | &#167; | параграф |
| © | &copy; | &#169; | знак copyright |
| « | &laquo; | &#171; | левая двойная угловая скобка |
|  | &shy; | &#173; | место возможного переноса |
| ® | &reg; | &#174; | знак зарегистрированной торговой марки |
| ° | &deg; | &#176; | градус |
| ² | &sup2; | &#178; | верхний индекс два квадрат |
| ³ | &sup3; | &#179; | верхний индекс три куб |
| · | &middot; | &#183; | точка по середине |
| » | &raquo; | &#187; | правая двойная угловая скобка |
| ½ | &frac12; | &#189; | дробь – одна вторая |
| × | &times; | &#215; | знак умножения |
| ÷ | &divide; | &#247; | знак деления |
| σ | &Sigma; | &#931; | греческая заглавная буква сигма |
| λ | &lambda; | &#955; | греческая строчная буква лямбда |
| μ | &mu; | &#956; | греческая строчная буква мю |
| • | &bull; | &#8226; | bullet - диск |
| … | &hellip; | &#8230; | многоточие ... |
| € | &euro; | &#8364; | валюта евро |

**Структура HTML-документа**

<HTML>

<HEAD><TITLE>Моя домашняя страница</TITLE></HEAD>

<BODY>

</BODY>

</HTML>

Первый тег, который вы здесь видите, **<**HTML**>**, сообщает браузеру о том, что он имеет дело именно с документом в формате HTML. Тег **<**HTML**>** и парный ему закрывающий тег **</**HTML**>** можно считать как бы "конвертом", в который помещается весь документ. Любой HTML-документ состоит из заголовка, который задается при помощи тега **<**TITLE**>**, и тела документа, который определяется тегом **<**BODY**>**. В заголовке документа размещается служебная информация, комментарии автора и заголовок страницы, заключаемый в теги **<**TITLE**>**. Заголовок, вписанный между тегами **<**TITLE**>**, в основное текстовое поле браузеру не попадает, а, как правило, размещается в заголовке окна браузера. HTML-документ – это просто текстовый файл с расширением \*.htm(Unix-системы могут содержать файлы с расширением\*.html).Большинство элементов языка HTML описывает части содержания документа и помещается между тегами **<**BODY**>** и **</**BODY**>**, то есть, внутрь структурного элемента BODY. Такие элементы делят на блочные и текстовые. Блочные элементы относятся к частям текста уровня абзаца. Текстовые элементы описывают свойства отдельных фраз и еще более мелких частей текста. Теперь можно сформулировать правила вложения элементов.

1. Элементы не должны пересекаться. Другими словами, если открывающий тег располагается внутри элемента, то и соответствующий закрывающий тег должен располагаться внутри этого же элемента.

2. Блочные элементы могут содержать вложенные блочные и текстовые элементы.

3. Текстовые элементы могут содержать вложенные текстовые элементы.

4. Текстовые элементы не могут содержать вложенные блочные элементы. Строго говоря, все правила языка HTML можно рассматривать исключительно как «пожелания». Средство, используемое для отображения Web-документа, сделает все возможное, чтобы истолковать разметку наиболее разумным образом. Тем не менее, гарантию правильного воспроизведения документа дает только неукоснительное следование требованиям спецификации языка.

**Функциональные блочные элементы**

В большинстве документов основными функциональными элементами являются заголовки и абзацы. Язык HTML поддерживает шесть уровней заголовков. Они задаются при помощи парных тегов от **<**Н1**>** до **<**Н6**>**. При отображении Web-документа на экране компьютера эти элементы показываются при помощи шрифтов разного размера.

Обычные абзацы задаются с помощью парного тега **<**Р**>**. Язык HTML не содержит средств для создания абзацного отступа («красной строки»), поэтому при отображении на экране компьютера абзацы разделяются пустой строкой. Закрывающий тег **</**Р**>** рассматривается как необязательный. Подразумевается, что он стоит перед тегом, который задает начало очередного абзаца документа.

Например

<h1>Заголовок</h1>

<p>Первый абзац

<p>Второй абзац

<h2>Заголовок второго уровня</h2>

Реализация примера

**Заголовок**

Первый абзац

Второй абзац

**Заголовок второго уровня**

Следствием наличия специального тега, определяющего абзац, является тот факт, что обычного символа конца строки, вводимого по нажатию клавиши ENTER, для создания абзацного отступа недостаточно. Язык HTML рассматривает символы конца строки и пробелы особым образом. Любая последовательность, состоящая только из пробелов и символов конца строки, при отображении документа рассматривается как одиночный пробел. Это, в частности, означает, что символ конца строки даже не осуществляет перехода на новую строку (для этой цели используется текстовый элемент, задаваемый непарным тегом **<**BR**>**).

**Лабораторная работа №1.  
Текстовое оформление страниц**

С помощью тега **<**font**>** можно изменить параметры шрифта. Для тега используются следующие параметры: face, size и color.

Параметр Face служит для задания гарнитуры шрифтов использующихся для текста. Названий шрифтов можно указать несколько, через запятую. В этом случае, если первый указанный шрифт не будет найден, будет использоваться следующий по списку.

Пример 1. Использование параметра face

<font face="Arial, Helvetica, sans-serif">Текст будет написан шрифтом Arial.</font>

Sizeзадает размер шрифта в условных единицах от1до7.Средний размер, используемый по умолчанию равен 3. Размер шрифта можно указывать как абсолютной величиной (например , size=4), так и относительной (например, size=+1, size=-1). В последнем случае размер изменяется относительно базового.

Пример 2. Задание размера шрифта

<font size=1>Шрифт размера 1</font><br>

<font size=2>Шрифт размера 2</font><br>

<font size=3>Шрифт размера 3</font><br>

<font size=4>Шрифт размера 4</font><br>

<font size=5>Шрифт размера 5</font><br>

<font size=6>Шрифт размера 6</font><br>

<font size=7>Шрифт размера 7</font><br>

Реализация примера 2

Шрифт размера 1

Шрифт размера 2

Шрифт размера 3

Шрифт размера 4

Шрифт размера 5

Шрифт размера 6

Шрифт размера 7

Colorопределяет цвет текста,который можно задавать с помощью названий цветов или в шестнадцатеричном формате.

Пример 3. Изменение цвета текста

<font size=5 color=redface=Arial>П</font>ервая буква этого предложения будет написана шрифтом <span style=”background-color:red;”>Arial</span>, красным цветом и увеличенной.</font>

Реализация примера 3

Первая буква этого предложения будет написана шрифтом Arial, красным цветом и увеличенной.

Видоизменение текста – средства его форматирования, такие как выбор начертания шрифта и использование эффектов, позволяющих менять вид текста. В таблице перечислены основные теги, которые применяются для изменения оформления текста.

|  |  |  |
| --- | --- | --- |
| **Код HTML** | **Описание** | **Пример** |
| <b>Текст</b> | Жирный текст | **Текст** |
| <strong>Текст</strong> | Жирный текст | Текст |
| <i>Текст</i> | Курсивный текст | *Текст* |
| <em>Текст</em> | Курсивный текст | *Текст* |
| <u>Текст</u> | Подчеркнутый текст | Текст |
| <sup>Текст</sup> | Верхний индекс | e=mc2 |
| <sub>Текст</sub> | Нижний индекс | e=mc2 |
| <strike>Текст</strike> | Зачеркнутый текст | ~~Текст~~ |
| <pre>Текст</pre> | Текст пишется как есть, включая все  пробелы | Текст |

Обычно для создания верхнего или нижнего индекса используется тег small,делающий индекс меньше по размеру основного шрифта.

Пример 4. Создание нижнего индекса

<b>Формула серной кислоты: </b><i>H<sub><small>2</small></sub>SO<sub><small>4</small></sub></i>

Реализация примера 4

**Формула серной кислоты:** *H2SO4*

Выравнивание текста определяет его внешний вид и ориентацию краев абзаца и может выполняться по левому, правому краю, по центру или по ширине.

|  |  |  |
| --- | --- | --- |
| **Код HTML** | **Описание** | **Пример** |
| <p>Текст</p> | Добавляет новый параграф, по умолчанию выровненный по левому краю. Перед параграфом автоматически добавляется пустая строка. | Текст |
| <p align=left>Текст</p> | Выравнивание по левому краю. | Текст |
| <p align=right>Текст</p> | Выравнивание по правому краю. | Текст |
| <p align=center>Текст</p> | Выравнивание по центру. | Текст |
| <p align=justify>Текст</p> | Выравнивание по ширине. | Текст по ширине |

**Задание к выполнению лабораторной работы**

Оформите текст, как показано ниже (для прорисовки между абзацами линии используйте тег <hr>):

Язык HTML был разработан британским учёным **Тимом Бернерсом-Ли** приблизительно в 1986-1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. *HTML* создавался как язык для обмена научной и технической документацией, пригодный для использования *людьми*, не являющимися специалистами в области вёрстки.

HTML успешно справлялся с проблемой сложности ~~SGML~~ путём определения небольшого набора структурных и семантических элементов – дескрипторов. Дескрипторы также часто называют «**те***га*ми». С помощью *HTML* можно легко создать относительно простой, но красиво оформленный ~~документ~~.

Помимо упр~~още~~ния структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были ***добавлены*** позже. Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (~~отображения~~).

В идеале, текст с разметкой HTML должен был без стилистических и структурных ~~искажений~~ воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран *мобильного телефона* или ~~устройства~~ и программы голосового воспроизведения текстов).

Однако современное применение ~~HTML~~ очень далеко от его изначальной задачи. Например, тег <~~TABLE~~> предназначен для создания в документах таблиц, но часто используется и для оформления размещения элементов на странице. С течением времени основная идея платформонезависимости языка ~~HTML~~ была принесена в жертву современным потребностям в мультимедийном и графическом оформлении.

**Контрольные вопросы**

1. Для чего служит тег <font>?

2. В чем отличие между тегами <b> и <strong>? Какие плюсы и минусы.

3. В чем отличие между тегами <i> и <em>? Какие плюсы и минусы.

4. Для чего нужно задавать разные размеры текста относительно основного?

5. Чрезмерное использование различного форматирования текста может ли как-то повлиять на восприятие пользователя?

6. Какие Вы знаете еще способы задания размера шрифта?

**Лабораторная работа №2.  
Создание гиперссылок**

Для создания ссылки необходимо сообщить браузеру, что является ссылкой, а также указать адрес документа, на который следует сделать ссылку. Оба действия выполняются с помощью тега <A>, который имеет единственный параметр href. В качестве значения используется адрес документа (URL).

Адрес ссылки может быть абсолютным и относительным. Абсолютные адреса работают везде и всюду независимо от имени сайта или веб-страницы, где прописана ссылка.

Пример 1. Использование абсолютных ссылок

<html>

<body>

<a href=www.yandex.ru>Поисковая система Яндекс</a>

</body>

</html>

Относительные ссылки, как следует из их названия, построены относительно текущего документа или адреса. Примеры таких адресов:

1. /
2. /demo/
3. /images/pic.gif
4. ../help/me.html
5. manual/info.html

Первые две ссылки называются неполные и указывают веб-серверу загружать файл index.html (или default.html), находящемуся в корне сайта (пример 1) или папке demo (пример 2). Если файл index.html отсутствует, браузер, как правило, показывает список файлов, находящихся в данном каталоге. Слэш перед адресом говорит о том, что адресация начинается от корня сайта (пример 3), двоеточие – перейти на уровень выше в списке каталогов сайта (пример 4).

Пример 2. Использование относительных ссылок

<html>

<body>

<a href=images/xxx.jpg>Посмотрите на мою фотографию!</a><br>

<a href=tip.html>Как сделать такое же фото?</a>

</body>

</html>

**Ссылки внутри страницы**

Большие документы читаются лучше, если они имеют оглавление со ссылками на соответствующие разделы. Для создания ссылки следует вначале сделать закладку в соответствующем месте и дать ей имя при помощи параметра name тега <А>.

Пример 3. Создание внутренней ссылки

<html>

<body>

<a name=top></a>Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986-1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов – дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже…

<a href=#top>Наверх</a>

</body>

</html>

Между тегами **<**a name=top> и </a**>** отсутствует текст, так как требуется лишь указать местоположение перехода по ссылке, находящейся внизу страницы. Имя ссылки на закладку начинается символом **#**, после чего идет название закладки. Название выбирается любое, соответствующее тематике.

Также можно делать ссылку на закладку, которая находится на другой веб-странице и даже другом сайте. Для этого в адресе ссылки надлежит указать ее адрес и в конце добавить символ решетки **#** и имя закладки.

Пример 4. Ссылка на закладку из другой веб-страницы

<html>

<body>

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

</body>

</html>

**Ссылка на новое окно**

Если требуется сделать ссылку на документ, который открывается в новом окне браузера, используется параметр target=\_blank тега <А>.

Создание нового окна обычно требуется в случаях, когда делается ссылка на другой сайт, в остальном лучше открывать документы в текущем окне, поскольку обилие окон может сбить читателя с толку.

Так как ссылки на текущее или новое окно ничем не отличаются друг от друга, на некоторых сайтах рядом со ссылкой ставят специальную иконку, показывающую, что документ открывается в новом окне.

Пример 5. Создание ссылки на новое окно

<html>

<body>

<a href=”cubook.supernew.org”>Обычная ссылка на сайт cubook.supernew.org</a><br>



<a href=”cubook.supernew.org” target=\_blank>Ссылка открывает новое окно на сайт cubook.supernew.org</a>

</body>

</html>

**Задание к выполнению лабораторной работы**

Используя внутренние ссылки, создайте следующий словарь терминов:

**Словарь терминов**

A B C D E F

А

<A>

Тег <a> является одним из важных элементов HTML и предназначен для создания ссылок. В зависимости от присутствия атрибутов name или href тег <a> устанавливает ссылку или якорь. Якорем называется закладка внутри страницы, которую можно указать в качестве цели ссылки. При использовании ссылки, которая указывает на якорь, происходит переход к закладке внутри веб-страницы.

<AREA>

Каждый элемент <area> определяет активные области изображения, которые являются ссылками. Рисунок с привязанными к нему активными областями называется в совокупности картой-изображением. Такая карта по внешнему виду ничем не отличается от обычного изображения, но при этом оно может быть разбито на невидимые зоны разной формы, где каждая из областей служит ссылкой.

В начало

B

<B>

Тег <b> устанавливает жирное начертание шрифта. Допустимо использовать этот тег совместно с другими тегами, которые определяют начертание текста.

<BODY>

Элемент <body> предназначен для хранения содержания веб-страницы (контента), отображаемого в окне браузера. Информацию, которую следует выводить в документе, следует располагать именно внутри контейнера <body>. К такой информации относится текст, изображения, теги, скрипты JavaScript и т.д.

Тег <body> также применяется для определения цветов ссылок и текста на веб-странице. Подобная практика в HTML 4 осуждается и взамен для указания цветовой схемы рекомендуется использовать стили, применяя их к селектору BODY.

<BR>

Тег <br> устанавливает перевод строки в том месте, где этот тег находится. В отличие от тега абзаца <p>, использование тега <br> не добавляет пустой отступ перед строкой. Если текст, в котором используется перевод строки, обтекает плавающий элемент, то с помощью атрибута clear тега <br> можно сделать так, чтобы следующая строка начиналась ниже элемента.

В начало

C

<CENTER>

Тег <center> выравнивает содержимое контейнера по центру относительно родительского элемента.

В начало

D

<DIV>

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

В начало

E

<EM>

Тег <em> предназначен для акцентирования текста. Браузеры отображают такой текст курсивным начертанием.

В начало

F

<FONT>

Тег <font> представляет собой контейнер для изменения характеристик шрифта, таких как размер, цвет и гарнитура. Хотя этот тег до сих пор поддерживается всеми браузерами, он считается устаревшим и от его использования рекомендуется отказаться в пользу стилей.

В начало

**Контрольные вопросы**

1. Что такое абсолютная ссылка? Какие ее основные функции?

2. Что такое относительная ссылка? Какие ее основные функции?

3. Чем отличается внешняя ссылка от внутренней?

4. Какие основные атрибуты должны быть у тега <a>?

5. Есть две ссылки <a name=top></a> и <a href=#top></a>. Какая из них верно написана? Обоснуйте свой ответ.

6. Можно ли использовать ссылку без закрывающего тега? Обоснуйте свой ответ.

**Лабораторная работа №3.  
Создание списков**

**Нумерованные списки**

Нумерованные списки представляют собой набор элементов с их порядковыми номерами. Вид и тип нумерации зависит от параметров тега <OL>, который и используется для создания списка. В качестве маркеров могут быть следующие значения: арабские цифры, заглавные латинские буквы, прописные латинские буквы, заглавные римские цифры, прописные римские цифры и т.д.

Ниже, в таблице приведены различные параметры тега <OL> и результат их применения.

|  |  |
| --- | --- |
| **Код HTML** | **Пример** |
| <ol>  <li>текст</li>  <li>текст</li>  <li>текст</li>  </ol> | Нумерованный список с параметрами по умолчанию:  1. текст  2. текст  3. текст |
| <ol start="5"> | Нумерованный список начинающийся с пяти:  5. текст  6. текст  7. текст |
| <ol type="A"> | Нумерованный список с заглавными буквами латинского  алфавита:  A. текст  B. текст  C. текст |
| <ol type="a"> | Нумерованный список с прописными буквами латинского  алфавита:  a текст  b текст  c текст |
| <ol type="I"> | Нумерованный список с римскими буквами:  I. текст  II. текст  III. текст |
| <ol type="i"> | Нумерованный список с прописными римскими буквами:  i. текст  ii. текст  iii. текст |
| <ol type="1"> | Нумерованный список с арабскими цифрами:  1. текст  2. текст  3. текст |
| <ol type="I" start="7"> | Список с римскими цифрами начинающийся с семи:  IV. текст  V. текст  VI. текст |

**Маркированные списки**

Маркированные списки позволяют разбить большой текст на отдельные блоки. Тем самым привлекается внимание читателя к тексту и повышается его читабельность. С учетом худшего восприятия текста с экрана монитора, чем печатного варианта, это является весьма полезным приемом.

Для установки маркированного списка используется тег <UL> и <LI>.

Пример 1. Создание маркированного списка

<html>

<head>

<body>

Что следует учитывать при тестировании сайта:

<ul>

<li>работоспособность всех ссылок</li>

<li>поддержку разных браузеров</li>

<li>читабельность текста</li>

</ul>

</body>

</html>

Реализация примера 1

Что следует учитывать при тестировании сайта:

* работоспособность всех ссылок
* поддержку разных браузеров
* читабельность текста

Обратите внимание, что у маркированного текста появляются отступы сверху и снизу. Чтобы от них избавиться, список можно делать без тега <UL>. При этом исчезнут и отступы текста перед маркерами.

Пример 2. Создание маркированного списка без отступов

<html>

<head>

<body>

Что следует учитывать при тестировании сайта:

<li>работоспособность всех ссылок</li>

<li>поддержку разных браузеров</li>

<li>читабельность текста</li>

</body>

</html>

Реализация примера 2

Что следует учитывать при тестировании сайта:

* работоспособность всех ссылок
* поддержку разных браузеров
* читабельность текста

Маркеры могут принимать один из трех видов: круг (по умолчанию), окружность и квадрат. Для выбора типа маркера используется параметр type="..." тега <UL>. Вместо многоточия подставляется одно из трех значений указанных в таблице.

|  |  |
| --- | --- |
| **Код HTML** | **Пример** |
| <ul type="disc"> | Что следует учитывать при тестировании сайта:   * работоспособность всех ссылок * поддержку разных браузеров * читабельность текста |
| <ul type="circle"> | Что следует учитывать при тестировании сайта:   * работоспособность всех ссылок * поддержку разных браузеров * читабельность текста |
| <ul type="square"> | Что следует учитывать при тестировании сайта:   * работоспособность всех ссылок * поддержку разных браузеров * читабельность текста |

С помощью CSS этот список можно расширить и вместо встроенных символов использовать в качестве маркера рисунок.

**Задание к выполнению лабораторной работы**

Создайте две web-страницы и оформите текст как представлено ниже:

Web-страница №1:

1. Оглянись, дружок, вокруг!
2. Вот компьютер – верный друг.
3. Он всегда тебе поможет:
4. Сложит, вычтет и умножит.
5. Наверху машины всей
6. Размещается дисплей.
7. Словно смелый капитан!
8. А на нем горит экран.
9. Ну а рядом главный блок:
10. Там бежит электроток
11. К самым важным микросхемам.
12. Этот блок зовут системным.
13. Он считает, он считает,
14. Сотни слов запоминает!
15. Он мыслитель хоть куда!
16. Только знаешь, в чем беда?
17. Выключаешь, все забудет,
18. Ничего он знать не будет –
19. Прямо как Рассеянный
20. С улицы Бассейной.
21. Кто ж ему поможет? Вот,
22. Познакомьтесь – дисковод!
23. Коль с компьютером ты дружен –
24. То и справочник не нужен!

Перейти на страницу №2

Web-страница №2:

* Поздравляю вас, друзья!
* Нынче весть услышал я:
* Твиттер русским стал, ей-богу!
* Если хочешь микроблогу
* Сделать русский интерфейс –
* Заходи в настройки смело
* И скорей берись за дело!
* Измени язык и сервис
* Выдаст блогов разных кучу
* На великом, на могучем…
* Твиттер стал сегодня круче!
* Не сказать ли тут «Ура!»?
* Мы теперь не лыком шиты,
* Мы в струе, в тенденции.
* Вмиг расширить русский сектор –
* В нашей компетенции!

Перейти на страницу №1

**Контрольные вопросы**

1. Какие типы маркированного списка Вы знаете?

2. Какие типы пронумерованного списка Вы знаете?

3. Для чего служит атрибут «start» в пронумерованном списке? Можно ли его использовать в маркированном списке?

4. Для чего служит атрибут «type» в маркированном списке? Можно ли его использовать в пронумерованном списке?

5. В каких моментах целесообразно использовать маркированный список, а когда пронумерованный?

**Лабораторная работа №4.  
Вставка изображений в html-страницы**

Для встраивания изображения в документ используется тег <IMG>, имеющий единственный обязательный параметр src, который определяет адрес файла с картинкой.

Файл с рисунком, изображенным ниже, называется sample.gif и размещается в папке images корня сайта.

Для указания адреса изображения можно задавать как абсолютный, так и относительный адрес.

Пример 1. Вставка изображения в документ

<html>

<body>

<img src="http://www.htmlbook/images/sample.gif"> - это абсолютный адрес размещения изображения

<img src="/images/sample.gif"> - адрес размещения изображения относительно корня сайта

<img src="images/sample.gif"> - адрес размещения изображения относительно текущего HTML-документа

</body>

</html>

**Выравнивание изображений**

Для изображений можно указывать их положение относительно текста или других изображений на веб-странице. Способ выравнивания изображений задается параметром align тега <IMG>. В таблице перечислены возможные значение этого параметра и результат его использования.

|  |  |  |
| --- | --- | --- |
| **Значение** | **Описание** | **Пример** |
| bottom | Выравнивание нижней границы изображения по окружающему тексту.. | Lorem ipsum dolor sit amet, consectetuer adipiscing elit... |
| left | Выравнивает изображение по левому краю окна. | Lorem ipsum dolor sit amet, consectetuer adipiscing elit... |
| middle | Выравнивание середины изображения по базовой линии текущей строки. | Lorem ipsum dolor sit amet, consectetuer adipiscing elit... |
| right | Выравнивает изображение по правому краю окна. | Lorem ipsum dolor sit amet, consectetuer adipiscing elit... |
| top | Верхняя граница изображения выравнивается по самому высокому элементу текущей строки. | Lorem ipsum dolor sit amet, consectetuer adipiscing elit... |

Наиболее популярные параметры – left и right, создающие обтекание текста вокруг изображения. Чтобы текст не прилегал плотно к рисунку, рекомендуется в теге IMG добавить параметр hspace и vspace, задающих расстояние до текста в пикселях.

Пример 2. Обтекание текста вокруг рисунка

<html>

<body>

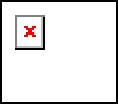
<img src="HLPBELL.GIF" width=50 height=50 hspace=10 vspace=10 align=left>В последние два десятилетия Интернет прочно вошел в нашу жизнь. В наше время он охватывает почти все сферы жизни: культура, искусство, бизнес. Наряду с обычными музеями, библиотеками, магазинами все большую популярность во всем мире завоевывают аналогичные Интернет-ресурсы.

</body>

</html>

Реализация примера 2

В последние два десятилетия Интернет прочно вошел в нашу жизнь. В наше время он охватывает почти все сферы жизни: культура, искусство, бизнес. Наряду с обычными музеями, библиотеками, магазинами все большую популярность во всем мире завоевывают аналогичные Интернет-ресурсы.



**Задание к выполнению лабораторной работы**

Выровняйте рисунки и текст так, как показано ниже:

|  |  |
| --- | --- |
|  | Этот рисунок находится слева и текст выравнивается по левой границе.  Эффект достигается без таблицы. |
|  | Этот рисунок находится слева и текст выравнивается по центру.  Эффект достигается без таблицы. |
| Этот рисунок находится справа и текст выравнивается по правой границе.  Эффект достигается без таблицы. |  |

**Контрольные вопросы**

1. Нужен ли закрывающий тег для тега <img> и почему?

2. Какие существуют адреса размещения изображений?

3. Есть ли разница в приведённых ниже адресах изображений?

<img src="/images/sample.gif"> и <img src="images/sample.gif">

4. Можно ли использовать тег <img> за пределами тега <body>?

**Лабораторная работа №5.  
Создание таблиц**

Таблица состоит из строк и столбцов ячеек, которые могут содержать текст и рисунки. Обычно таблицы используются для упорядочения и представления данных, однако возможности таблиц этим не ограничиваются. C помощью таблиц удобно верстать макеты страниц, расположив нужным образом фрагменты текста и изображений.

Для добавления таблицы на веб-страницу используется тег-контейнер <TABLE>. Таблица должна содержать хотя бы одну строку и колонку.

Для добавления строк используются теги <tr> и </tr>. Чтобы разделить строки на колонки применяются теги <td> и </td>.

**Параметры таблицы**

Для изменения вида и свойств таблицы используется множество параметров, которые добавляются в теге <TABLE>.

<table параметр1=... параметр2=...>

Описание параметров таблицы и их свойств описано ниже.

|  |  |  |  |
| --- | --- | --- | --- |
| **Свойство** | **Значение** | **Описание** | **Пример** |
| **align=** | Left  Right  Center | Выравнивание таблицы | align=center |
| **background=** | URL | Фоновый рисунок | background=pic.gif |
| **bgcolor=** | #rrggbb  red | Цвет фона таблицы | bgcolor=#FF9900  bgcolor=red |
| **border=** | Число | Толщина рамки в пикселах | border=2 |
| **bordercolor=** | #rrggbb  red | Цвет рамки | bordercolor=#  333333 |
| **bordercolordark=** | #rrggbb  red | Тень рамки | bordercolordark=#  f0f0f0 |
| **cellpadding=** | Число | Расстояние между ячейкой и ее содержимым | cellpadding=7 |
| **cellspacing=** | Число | Расстояние между ячейками | cellspacing=3 |
| **nowrap** |  | Запрещает переносы строк в тексте | <table nowrap> |
| **frame=** | Void  Above  Below  Lhs  Rhs  Hsides  Vsides  Box | Задание типа рамки таблицы | frame=hsides |
| **valign=** | Top  Bottom | Выравнивание по высоте | valign=top |
| **width=** | Число  Проценты % | Минимальная ширина таблицы, можно задавать в  пикселях или процентах | width=90% |
| **height** | Число  Проценты % | Минимальная высота таблицы, можно задавать в  пикселях или процентах | height=18 |

Примечание

1. Таблица, если не указано особо, всегда выравнивается по левому краю.

2. Параметр background, отвечающий за рисунок фона, своеобразно понимается в разных браузерах. IE вставляет картинку во всю таблицу, если таблица по размеру больше фонового рисунка, он повторяется по горизонтали или вертикали. Netscape добавляет фоновое изображение в каждую ячейку таблицы.

3. По умолчанию, таблица выводится без рамки. Однако Netscape добавляет тонкую линию между ячеек. Чтобы ее не было, всегда указывайте параметр border=0.

4. Если ширина таблицы не указана, то она подгоняется под содержание ячеек.

**Задание к выполнению лабораторной работы**

**1.1 Создание таблицы**

<HTML>

<HEAD>

<TITLE>ТАБЛИЦА</TITLE>

</HEAD>

<BODY>

<TABLE border="2" align="center">

<TR>

<TD colspan=2 align="center"><B>Заголовок Таблицы.</B>

</TD>

</TR>

<TR>

<TD align="center">Первая ячейка первой строки</TD>

<TD align="center">Вторая ячейка первой строки</TD>

</TR>

<TR>

<TD align="center">Первая ячейка второй строки</TD>

<TD align="center">Вторая ячейка второй строки</TD>

</TR>

</TABLE>

</BODY>

</HTML>

Тег <TABLE> задает таблицу.

Атрибуты border="2" и align="center" задают, соответственно, размер границ таблицы и выравнивание ее по центру страницы.

Тег <TR> задает строку таблицы.

Тег <TD> задает ячейку таблицы.

Атрибут colspan=n объединяет n ячеек по горизонтали (по столбцам).

В результате получится следующая таблица, состоящая из двух столбцов и двух строк:

|  |  |
| --- | --- |
| **Заголовок Таблицы.** | |
| 1 ячейка первой строки | 2 ячейка первой строки |
| 1 ячейка второй строки | 2 ячейка второй строки |

**1.2 Сложная таблица**

<HTML>

<HEAD>

<TITLE>СЛОЖНАЯ ТАБЛИЦА</TITLE>

</HEAD>

<BODY>

<TABLE border="1" width="75%" align="center"><TR>

<TD width="66%" colspan="2">

<P align="center">Две ячейки, объединенные по горизонтали

</TD>

</TR>

<TR>

<TD width="33%" rowspan="2" valign="middle">

Две ячейки, объединенные по вертикали

</TD>

<TD width="33%">по левому краю</TD>

</TR>

<TR>

<TD width="33%"><P align="right" >по правому краю </TD>

</TR>

</TABLE>

</BODY>

</HTML>

Атрибут colspan=n объединяет n ячеек по горизонтали (по столбцам).

Атрибут rowspan=n объединяет n ячеек по вертикали (по строкам).

Атрибут valign="middle" выравнивает текст в ячейке по центру ячейки по вертикали.

В результате получается следующая таблица:

|  |  |
| --- | --- |
| Две ячейки, объединенные по горизонтали | |
| Две ячейки, объединенные по вертикали | по левому краю |
| по правому краю |

**1.3 Более сложная таблица**

<HTML>

<HEAD>

<TITLE>УЧЕБНАЯ ТАБЛИЦА</TITLE>

</HEAD>

<BODY>

<TABLE WIDTH="50%" BORDER="6" CELLSPACING="6" CELLPADDING="20" ALIGN="center" BORDERCOLORLIGHT="Lime" BORDERCOLORDARK="Green" BGCOLOR="#DFFFDF">

<THEAD BGCOLOR="Aqua">

<TR><TH COLSPAN="3">УЧЕБНАЯ ТАБЛИЦА</TH></TR>

</THEAD>

<TBODY>

<TR>

<TD WIDTH="33%">Это первая ячейка</TD>

<TD WIDTH="33%">Это вторая ячейка</TD>

<TD ROWSPAN="3">А это три ячейки третьего столбца объединились в одну большую</TD>

</TR>

<TR>

<TD COLSPAN="2">Это единственная ячейка второй строки, объединяющая оба столбца</TD>

</TR>

<TR>

<TD>Это первая ячейка третьей строки</TD>

<TD WIDTH="33%">А это вторая ячейка третьей строки</TD> </TR>

</TBODY>

<TFOOT BGCOLOR="Yellow">

<TR>

<TD COLSPAN="3" ALIGN="center">

<SMALL>конец</SMALL>

</TD>

</TR>

</TFOOT>

</TABLE>

</BODY>

</HTML>

Атрибут CELLSPACING="6" задает свободное пространство между ячейками таблицы.

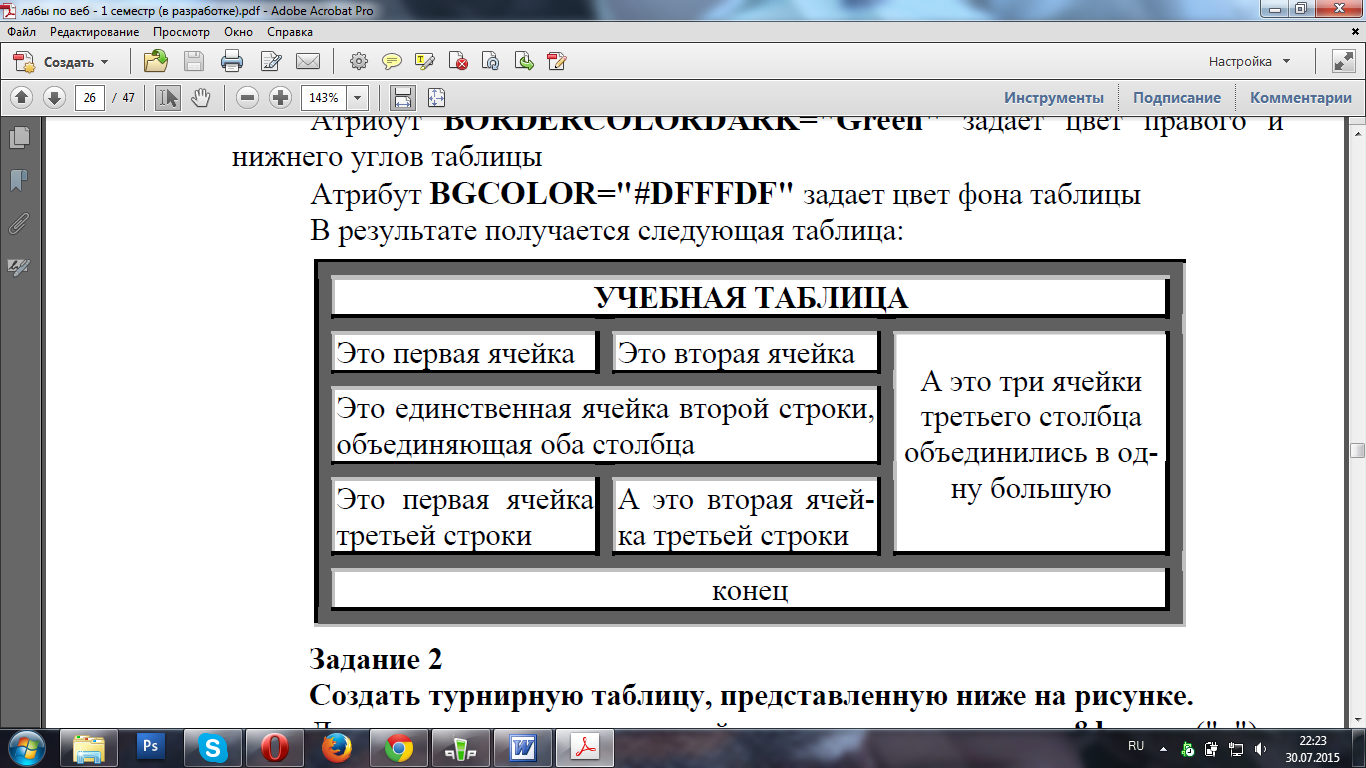
Атрибут CELLPADDING="20" задает свободное пространство между данными в ячейке и ее границами.

Атрибут BORDERCOLORLIGHT="Lime" задает цвет левого и верхнего углов таблицы.

Атрибут BORDERCOLORDARK="Green" задает цвет правого и нижнего углов таблицы.

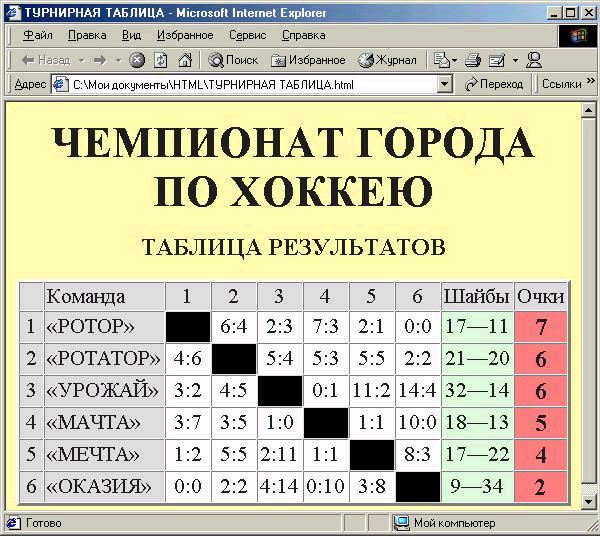
Атрибут BGCOLOR="#DFFFDF" задает цвет фона таблицы.

В результате получается следующая таблица:



**1.4 Создание таблицы результатов чемпионата по хоккею**

Освоив материал и получив определенные навыки, сделайте таблицу как приведено ниже:



Для задания кавычек используйте специальные символы &laquo; ("«") и &raquo; ("»"), для создания длинного тире используйте специальный символ &mdash; ("—").

**Контрольные вопросы**

1. Для чего служат теги <tr> и </tr>?

2. Для чего служат теги <td> и </td>?

3. Назовите 3-4 атрибута и их свойства, которые может использовать тег <table>.

4. Можно ли объединить две и более ячейки таблицы в одну?

**Лабораторная работа №6.  
Создание форм**

Форма HTML представляет собой документ, созданный с использованием элементов HTML. Назначением формы является сбор информации от пользователей. После того как пользователь заполнит форму и запускает процесс ее обработки, информация из нее попадает в программу, работающую на сервере. Другая программа под названием Common Gateway Interface (CGI) обрабатывает ее. Таким образом, пользователь может интерактивно взаимодействовать с сервером Web через Internet.

Элемент <FORM> используется для обозначения документа как формы. Данный элемент определяет границы использования других элементов, размещаемых в форме. <FORM> определяется последовательностью элементов <INPUT>, размещенных внутри пары <FORM> и </FORM>. Элемент формы использует как метод, так и действие для описания обработки формой данных, вводимых пользователем:

* Метод (GET или POST) – определяет, как программист должен обрабатывать входные данные из формы.
* Действие – указывает на URI программы, ответственной за обработку данных.

Данный элемент поддерживает атрибуты ACTION, ENCTYPE, METHOD.

Элемент <INPUT> используется для определения области внутри формы, где собираются данные. Данный элемент представляет собой поле для ввода информации пользователем (обычно одна строка текста). В этом случае требуется наличие атрибута NAME для определения наименования переменной поля. Можно использовать следующие атрибуты:

* MAXLENGTH – ограничивает число вводимых символов (по умолчанию ограничений нет).
* SIZE – размер видимой на экране области, занимаемой текущим полем. Если MAXLENGTH больше SIZE, браузер будет прокручивать данные в окне.
* VALUE – определяет начальное значение поля ввода.

А также атрибуты: ALIGN, CHECKED, NAME, SRC, TYPE.

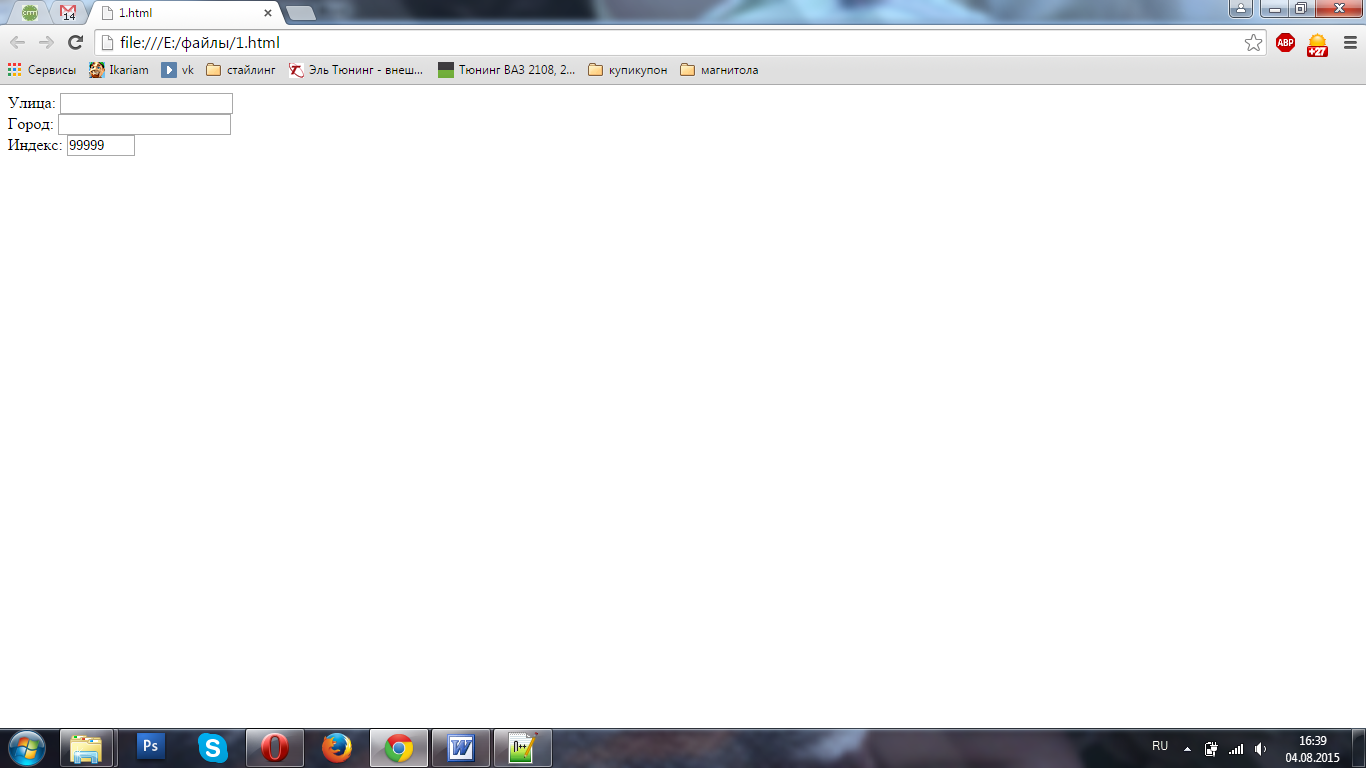
Пример 1. Простая форма для ввода

<P>Улица: <INPUT NAME="street"><BR>

Город: <INPUT NAME="city" SIZE="20" MAXLENGTH="20"><BR>

Индекс: <INPUT NAME="zip" SIZE="5" MAXLENGTH="5" VALUE="99999"><BR>

Реализация примера 1



При создании форм часто бывает необходимо получить ответ пользователя на вопрос типа (Да/Нет) или (Правда/Ложь). Например, нужно выбрать из списка несколько значений. Для создания независимых кнопок в формах можно использовать атрибут CHECKBOX. В зависимости от содержания можно отметить несколько флагов.

Вместе с атрибутом CHECKBOX должны использоваться следующие атрибуты:

* CHECKED – инициализировать данный флаг, как отмеченный.
* NAME – наименование поля ввода формы.
* VALUE – значение поля ввода.

Пример 2. Множественный выбор

<P>Выберите используемую версию HTML:

<FORM>

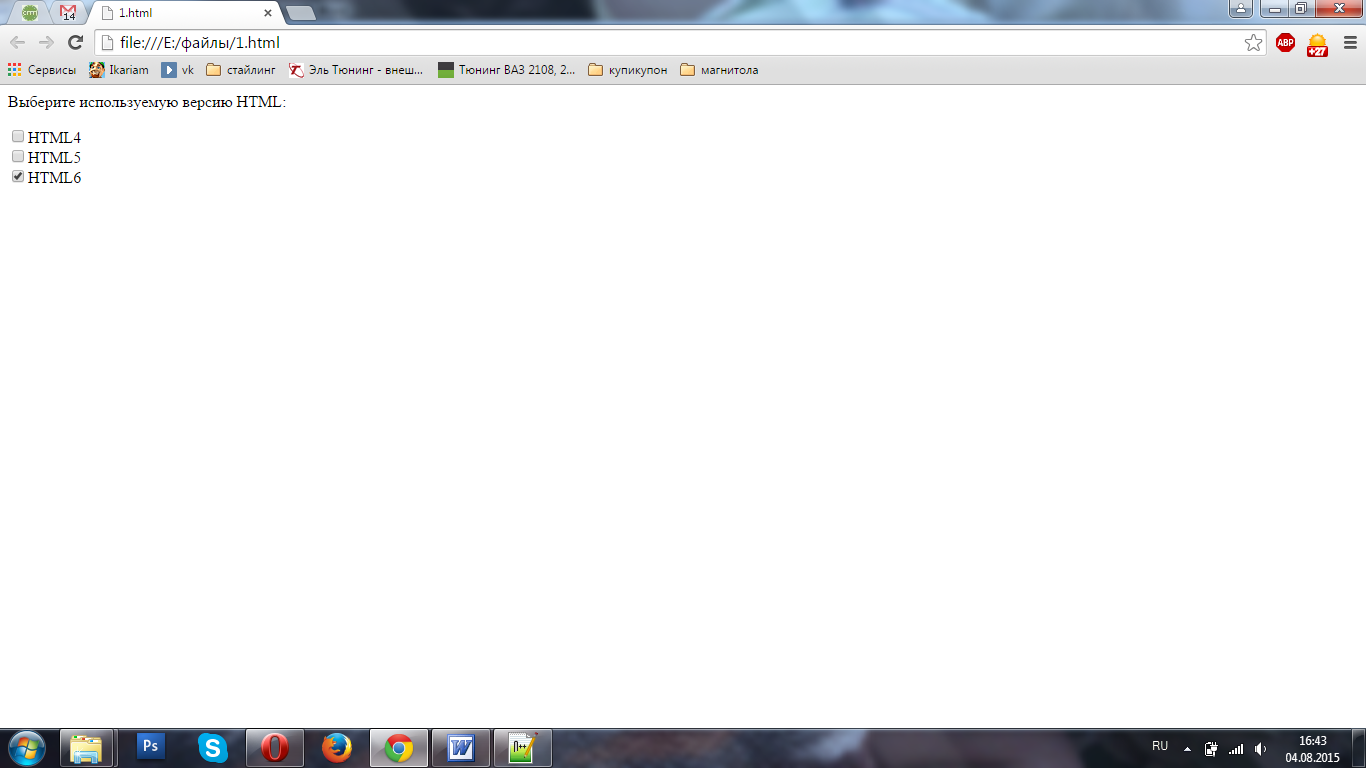
<INPUT TYPE="checkbox" NAME="html" VALUE="html4">HTML4<BR>

<INPUT TYPE="checkbox" NAME="html" VALUE="html5">HTML5<BR>

<INPUT TYPE="checkbox" NAME="html" VALUE="html6" CHECKED>HTML6<BR>

</FORM>

Реализация примера 2



В зависимости от содержимого формы может случиться так, что пользователю потребуется щелкнуть мышью на изображении, чтобы завершить работу с формой. Для организации этого используется атрибут IMAGE. После щелчка пользователя по изображению браузер сохраняет координаты соответствующей точки экрана и принимает всю форму.

Вместе с атрибутом IMAGE должны использоваться следующие атрибуты:

* ALIGN – необязательный атрибут, указывает на положение изображения на экране (аналогично элементу IMAGE).
* NAME – наименование поля ввода формы.
* SRC – URI файла - источника изображения.

Пример 3. Форма с изображением

<P>Выберите точку на изображении:

<INPUT TYPE="image" NAME="point" SRC="globe.gif">

Атрибут PASSWORD используется для организации ввода пароля без вывода на экран составляющих его символов (вместо символов выводятся звездочки).

Пример 4. Форма для ввода пароля

<FORM>

<P>Введите имя:

<INPUT NAME="login">

<P>Введите пароль:

<INPUT TYPE="password" NAME="p\_word">

</FORM>

Атрибут RADIO используется для организации выбора одного единственного варианта их нескольких возможных. Вместе с атрибутом RADIO должны использоваться следующие атрибуты:

* CHECKED – инициализировать данный флаг, как отмеченный.
* NAME – наименование поля ввода формы.
* VALUE – значение поля ввода.

Пример 5. Форма для выбора одного значения

<P>Выберите используемую версию CSS:

<FORM>

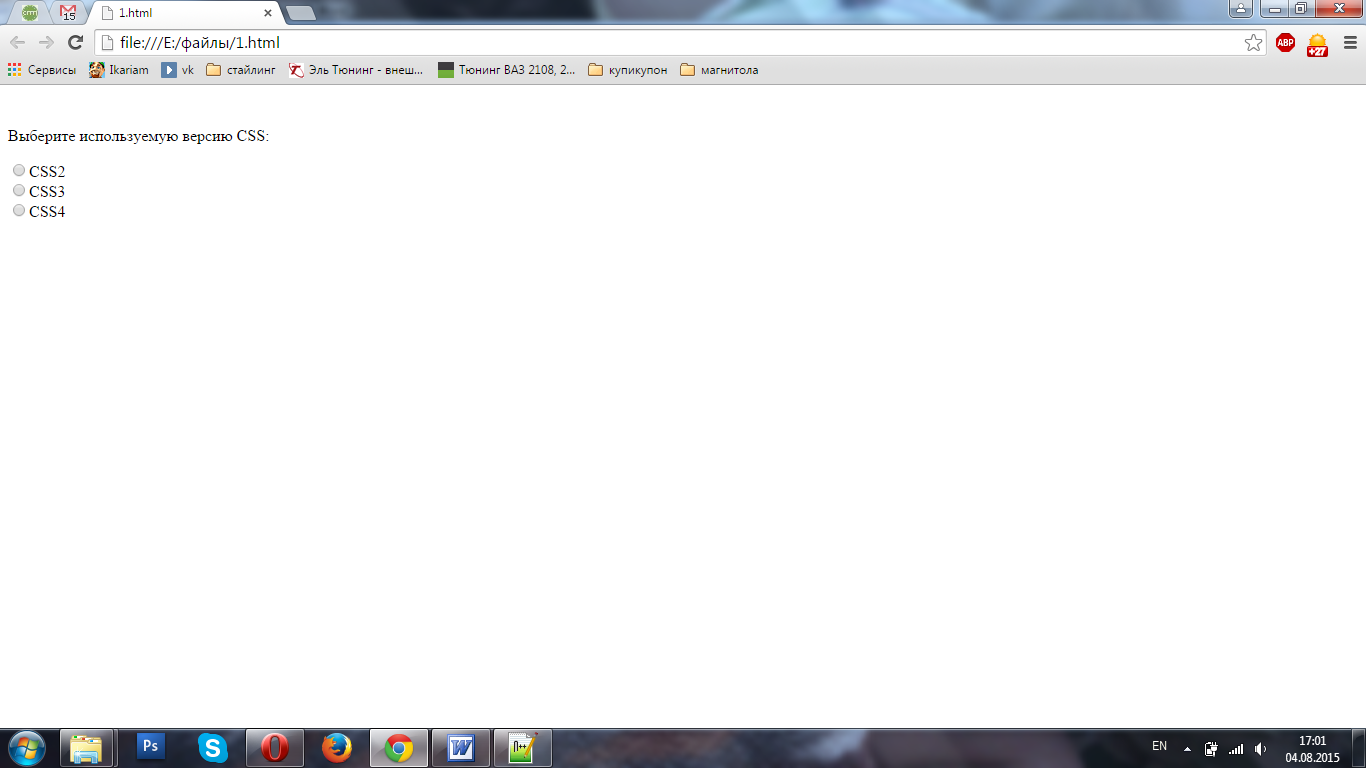
<INPUT TYPE="radio" NAME="css" VALUE="css2">CSS2<BR>

<INPUT TYPE="radio" NAME="css" VALUE="css3">CSS3<BR>

<INPUT TYPE="radio" NAME="css" VALUE="css4">CSS4<BR>

</FORM>

Реализация примера 5



Атрибут RESET используется для создания кнопки «Reset». При нажатии на эту кнопку форма восстанавливает первоначальные значения полей всех элементов <INPUT>, в которых присутствует атрибут RESET. Вместе с атрибутом RESET может использоваться атрибут.

Пример 6. Кнопка сброса

<P>

<FORM>

Код: <INPUT NAME="cod"><BR>

Телефон: <INPUT NAME="phone" SIZE="6" MAXLENGTH="6"><BR>

<P>

<INPUT TYPE=RESET>

</FORM>

Для организации списков с прокруткой и выпадающим меню можно использовать атрибут <SELECT>. Для определения списка пунктов используются элементы <OPTION> внутри <SELECT>. Вместе с атрибутом SELECT можно использовать следующие атрибуты:

* NAME – наименование объекта.
* MULTIPLE – позволяет выбрать более чем одно наименование.
* SIZE – определяет число пунктов, видимых для пользователя.
* SIZE=1 – браузер выводит список на экран в виде выпадающего меню (видно одно наименование).
* SIZE>1 – браузер представляет на экране обычный список (число – количество видимых наименований).
* C элементом OPTION можно использовать следующие атрибуты:
* SELECTED – для первоначального выбора значения элемента по умолчанию.
* VALUE – значение, возвращаемое формой после выбора пользователем данного пункта. По умолчанию значение поля равно элементу <OPTION>.

Когда пользователь заполняет форму, атрибут NAME элемента <SELECT> состыковывается с атрибутом VALUE элемента для формирования наименования, выбранного пользователем.

Пример 7. Создание списка

<FORM>

<SELECT NAME="Фрукты" SIZE=3>

<OPTION>HTML

<OPTION>CSS

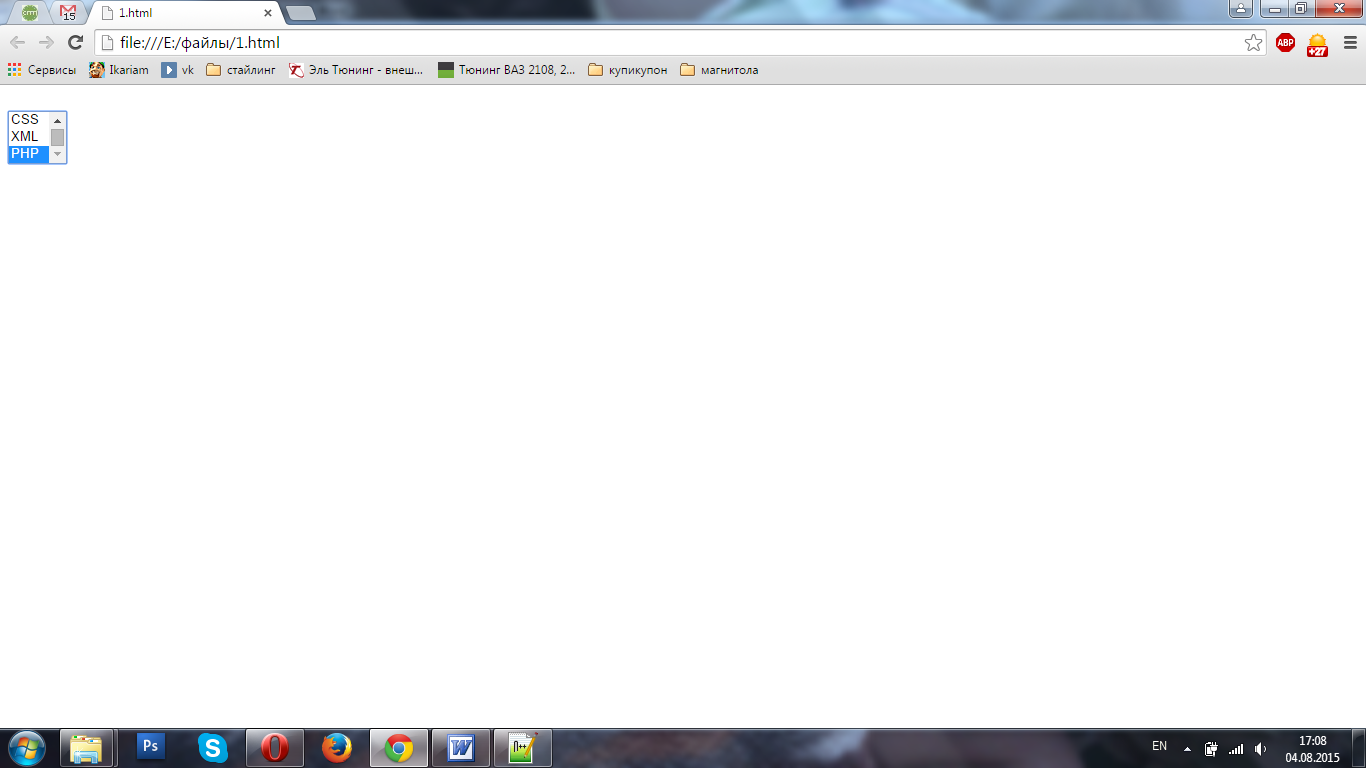
<OPTION value="xml">XML

<OPTION selected>PHP

</SELECT>

</FORM>

Реализация примера 7



Атрибут SUBMIT используется при окончании ввода пользователем данных. Браузер, в свою очередь выводит данный элемент, как кнопку, на которой пользователь может щелкнуть, чтобы завершить процесс редактирования. Вместе с атрибутом SUBMIT можно использовать следующие атрибуты:

* NAME – наименование кнопки SUBMIT.
* VALUE – значение переменной поля в вашей форме.

Пример 8. Отправка данных

<P>

<FORM>

Код: <INPUT NAME="cod"><BR>

Телефон: <INPUT NAME="phone" SIZE="6" MAXLENGTH="6"><BR>

<P>

<INPUT TYPE=RESET><INPUT TYPE=SUBMIT>

</FORM>

Атрибут TEXTAREA используется для ввода большого количества текстовой информации (несколько строк). Вместе с атрибутом TEXTAREA можно использовать следующие атрибуты:

* NAME – наименование поля.
* COLS – число колонок (символов) в текстовой области.
* ROWS – число видимых строк в текстовой области.

Пример 9. Создание поля для ввода текста

<FORM>

<TEXTAREA NAME="adress" COLS=64 ROWS=6>

Воронежская область

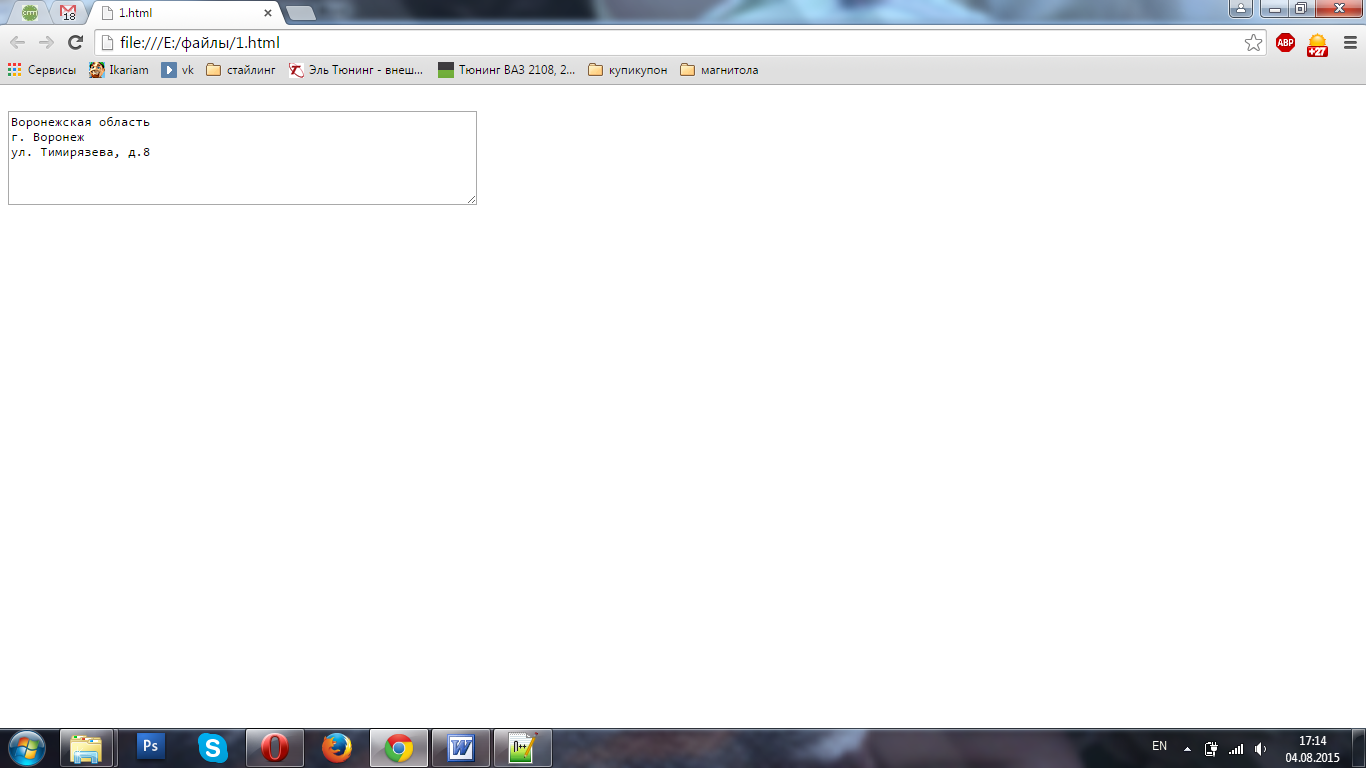
г. Воронеж

ул. Тимирязева, д.8

</TEXTAREA>

</FORM>

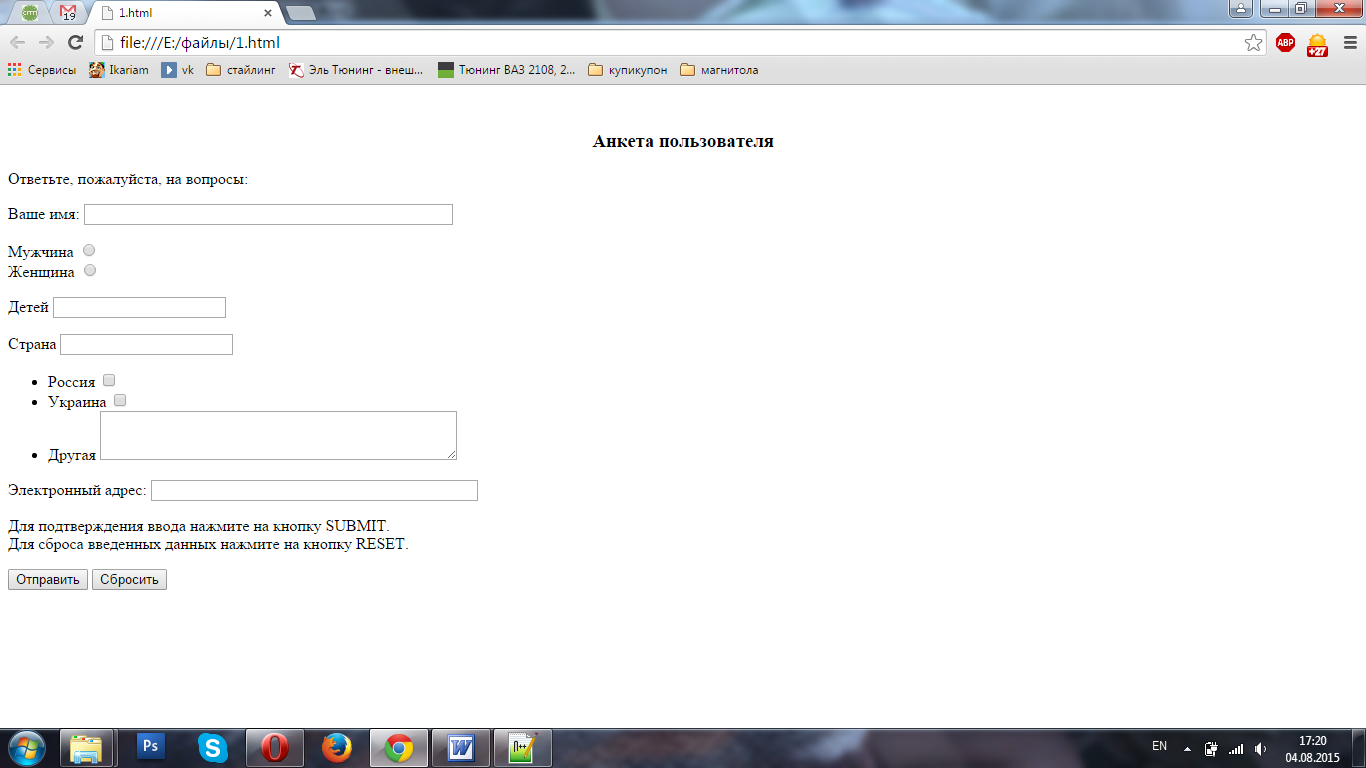
Реализация примера 9



**Задание к выполнению лабораторной работы**

Освоив материал и получив определенные навыки, создайте web-страницу, содержащую следующую информацию:

Анкета пользователя



**Контрольные вопросы**

1. Для чего служит тег <form>? Какие типы отправки данных в нём существуют?

2. Для чего служит тег <input>? Назовите пару типов для этого тега и их свойства.

3. Возможно ли сделать так, чтобы в текстовое поле вводилось вместо текста звёздочки?

4. В чём отличие между типами submit и button, ведь в браузере они отображаются в виде одинаковых кнопок?

**Лабораторная работа №7.  
Создание фреймов**

Фреймы – это окна независимого просмотра HTML-документов. Иногда бывает очень удобно использовать навигацию по странице в виде меню, оформленного в отдельном фрейме, и основного окна, где будет представлена вся основная информация, определяемая действиями пользователя в области меню.

Для создания фрейма используется тег FRAMESET, который заменяет тег BODY в документе и используется для разделения экрана на области. Внутри данного тега находятся теги FRAME, которые указывают на HTML-документ, предназначенный для загрузки в область.

Пример 1. Создание простого фрейма

<HTML>

<FRAMESET cols="30%, 70%" border=0>

<FRAME src="menu.html" name="Menu">

<FRAME src="main.html" name="Main">

</FRAMESET>

</HTML>

В приведенном примере присутствует объявление двух фреймов, которые будут располагаться вертикальными полосами и занимать соответственно 30 и 70 процентов рабочей области. Вертикальное расположение устанавливается параметром cols="…", а для горизонтальных полос используют параметр rows="…". Параметр border="…" определяет границу между фреймами. Определение документа, изначально загружаемого при открытии этого файла (этот документ является HTML-страницей.), задается атрибутом тега <FRAME> src="…", при этом необходимо указать параметр name="…", позволяющий задать «имя» созданной области в виде последовательности латинских букв и цифр, использованной как значение этого атрибута. Это имя можно использовать, чтобы загружать новые документы в ранее созданную область. Для этого в тег <A>, определяющий гиперссылку, необходимо добавить атрибут target="…", значение которого совпадает с ранее определенным именем области. При переходе по данной гиперссылке новый документ загрузится в указанный фрейм. Например, предположим, что начальная страница Web-узла состоит из двух фреймов: слева располагается навигационная панель, а справа – текущая страница. Если правой области присвоено имя, используемое во всех ссылках, имеющихся в левой области, то щелчок на любой ссылке навигационной панели приведет к обновлению информации в соседней области, оставляя навигационную панель без изменений. В данном документе находятся только ссылки на другие (существующие) HTML-документы, которые будут загружены браузером при открытии страницы с фреймами. Файл menu.html имеет следующую структуру:

<HTML>

<HEAD>

<TITLE></TITLE>

</HEAD>

<BODY>

<H1>Первый фрейм</H1>

</BODY>

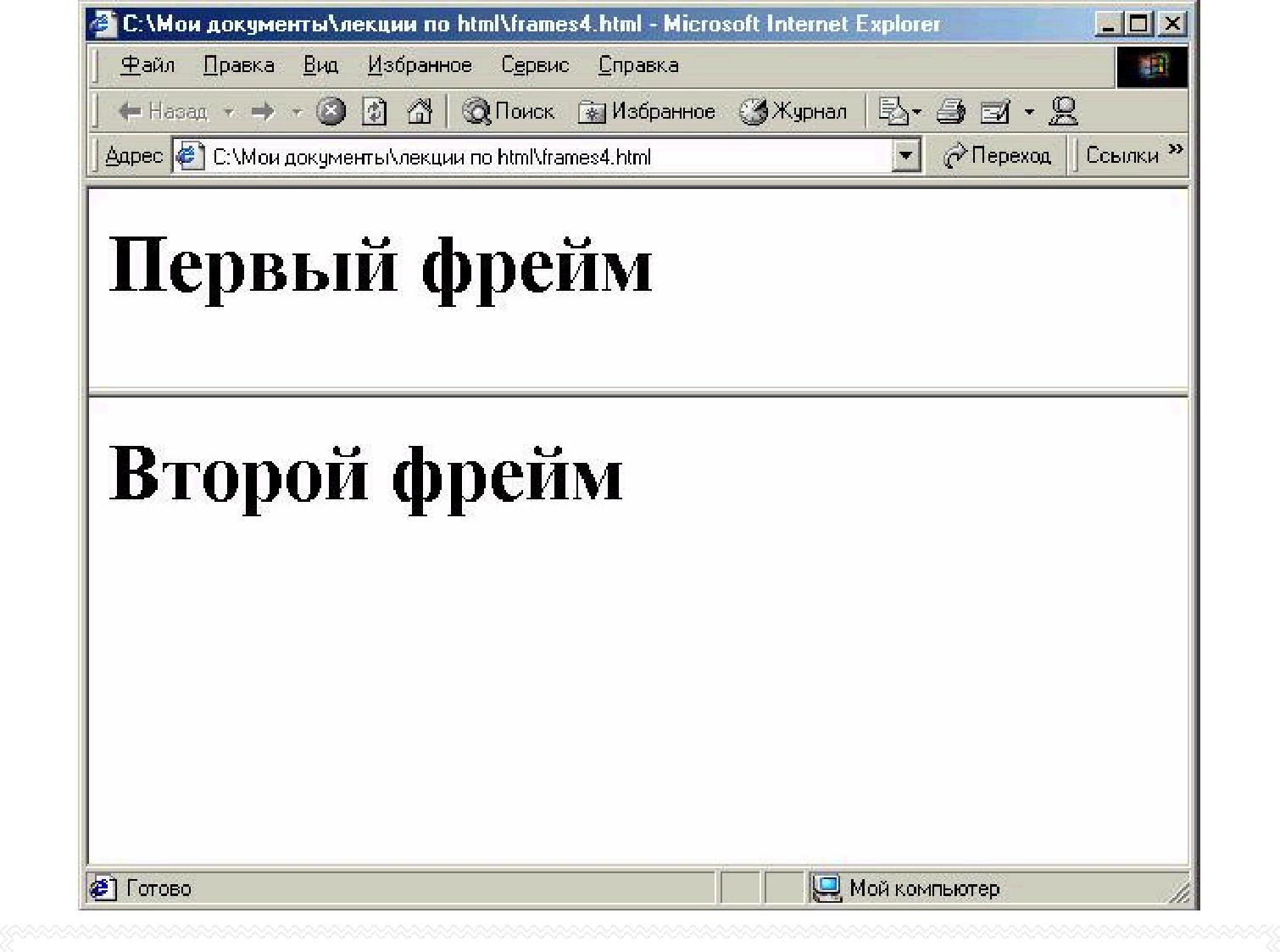
</HTML>

Файл main.html также имеет указанную выше структуру. Следует сказать, что оба этих файла должны располагаться в том же каталоге, где расположен файл с фреймами.

**Задание к выполнению лабораторной работы**

**1.1 Создание простейших фреймов**

Расположить эти фреймы по вертикали, используя атрибут rows="…". В результате у вас должно получиться следующее:



**1.2 Создание сложных фреймов**

Распространенное явление - комбинация вертикальных и горизонтальных фреймов. Символ \* означает все оставшееся место.

<FRAMESET cols="\*, 55%">

<FRAME src="homepage.htm" name="Frame1">

<FRAMESET rows="15%, 15%, 70%">

<FRAME src="menu.htm" name=" Frame2">

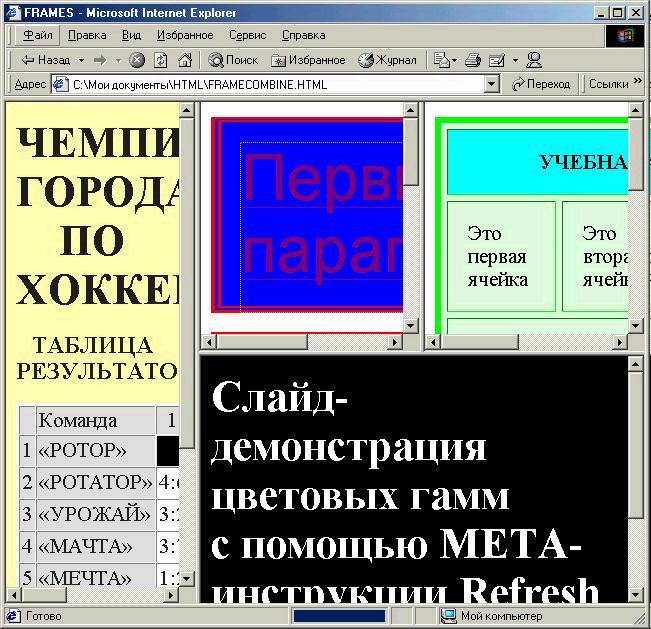
<FRAME src="menu2.htm" name=" Frame3">

<FRAME src="menu3.htm" name=" Frame4">

</FRAMESET>

</FRAMESET>

Используя вложенные теги FRAMESET, разбейте страницу следующим образом:



**1.3 Создание гиперссылок внутри фреймов**

Используя атрибут target="…", сделать во фреймах гиперссылки, по нажатию на которые в эти же или в другие фреймы произойдет загрузка других страниц.

**Контрольные вопросы**

1. Для чего нужны фреймы?

2. Корректен ли приведённый ниже код? Обоснуйте свой ответ.

<HTML>

<BODY>

<FRAMESET cols="45%, \*">

<FRAME src="menu.html" name="Menu">

<FRAME src="main.html" name="Main">

<FRAME src="above.html" name="Above">

</FRAMESET>

</BODY>

</HTML>

3. В каких размерах может измеряться фреймы?

4. Как можно быстро вычислить размерность последнего фрейма, если их много?

**Лабораторная работа №8.  
Каскадные таблицы стилей CSS**

С помощью CSS можно определять стиль и вид текста. Аналогично тому, что используется тег FONT, задающий свойства шрифта, но стили обладают большими возможностями и позволяют сократить код HTML.

Свойства шрифта

|  |  |  |  |
| --- | --- | --- | --- |
| **Свойство** | **Значение** | **Описание** | **Пример** |
| **font-family** | имя шрифта | Задает список шрифтов | p {font-family: Arial,  serif} |
| **font-style** | Normal  Italic  Oblique | Нормальный шрифт  Курсив  Наклонный шрифт | p {font-style: italic} |
| **fontvariant** | Norma  Small-caps | Капитель  (особые пропис-  ные буквы) | p {font-variant: smallcaps} |
| **font-weight** | Normal  Lighter  Bold  Bolder  100-900 | Нормальная жирность  Светлое начертание  Полужирный  Жирный  100-светлый шрифт,  900-самый жирный | p {font-weight: bold} |
| **font-size** | Размер шрифта  Normal  Pt  Px  % | Нормальный размер  Пункты  Пикселы  Проценты | font-size: normal  font-size: 12pt  font-size: 12px  font-size: 120% |

Когда размер шрифта задается абсолютными значениями, т.е. указывается конкретное значение шрифта в пунктах или пикселях, то изменить эту величину с помощью опции браузера невозможно. Если шрифт установлен слишком мелким, то исправить этот недостаток читателю простыми средствами не представляется возможным. Поэтому лучше задавать размер шрифта в процентах.

Пример 1. Задание свойств шрифта с помощью CSS

<html>

<style>

H1{font-family: Arial, Helvetica, Verdana, sans-serif; font-size: 150%; font-weight: light}

</style>

<body>

<H1>Заголовок</H1>

Обычный текст

</body>

</html>

Реализация примера 1

Заголовок

Обычный текст

**Свойства текста**

Кроме изменения параметров шрифтов, можно управлять и свойствами всего текста. Значения свойств приведены в таблице.

|  |  |  |  |
| --- | --- | --- | --- |
| **Свойство** | **Значение** | **Описание** | **Пример** |
| **line-height** | Normal  Множитель  Точно  % | Интерлиньяж  (межстрочный  интервал) | line-height: normal  line-height: 1.5  line-height: 12px  line-height: 120% |
| **text-decoration** | None  Underline  Overline  line-through  Blink | Убрать все оформление  Подчеркивание  Линия над текстом  Перечеркивание  Мигание текста | text-decoration: none |
| **text-transform** | None  Capitalize  Uppercase  Lowercase | Убрать все эффекты  Начинать С Прописных  ВСЕ ПРОПИСНЫЕ  все строчные | text-transform: capitalize |
| **text-align** | Left  Right  Center  Justify | Выравнивание текста | text-align: justify  выравнивание по ширине |
| **text-indent** | Точно  % | Отступ первой строки | text-indent:15px;  text-indent:10% |

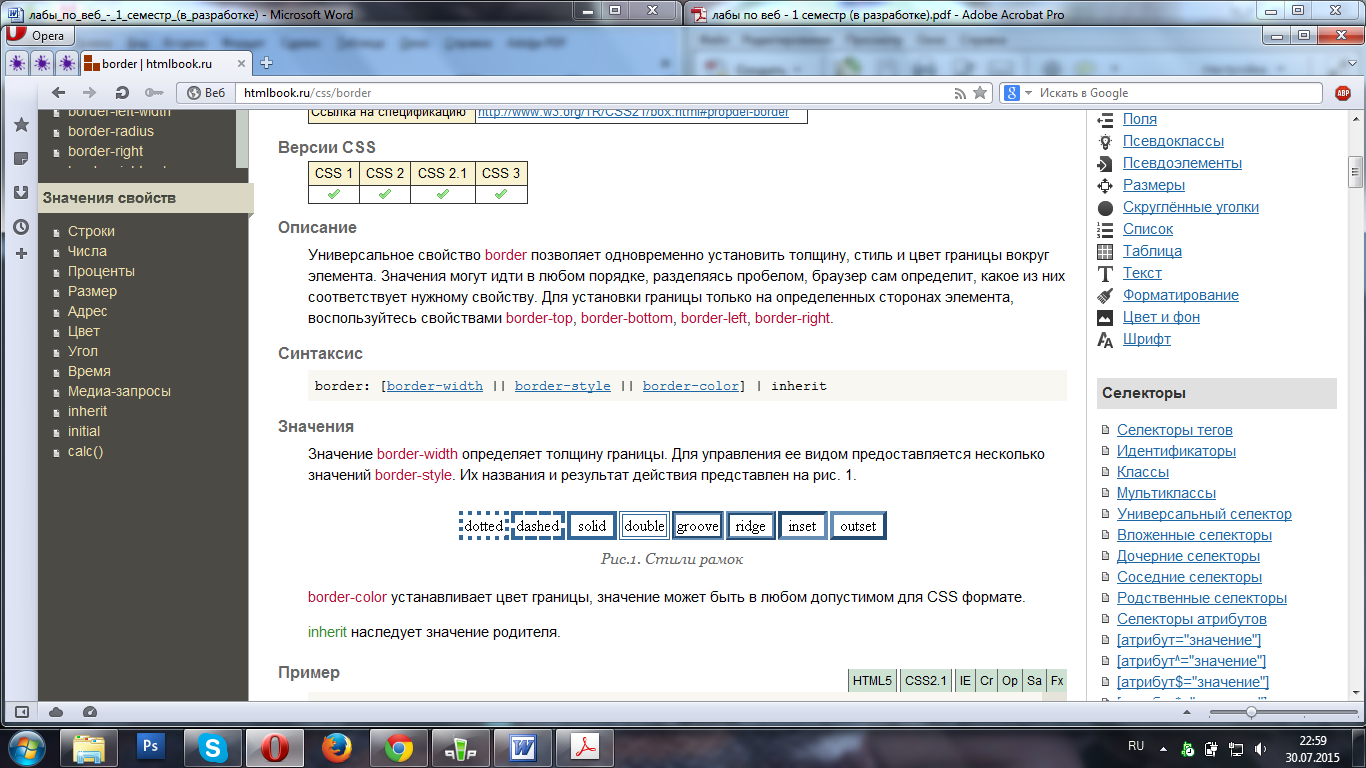
**Границы и рамки**

Спецификация CSS описывает несколько свойств, с помощью которых можно создавать границу вокруг различных элементов и управлять ее видом. Границы – одна из наиболее слабых сторон CSS, так как браузеры содержат большое количество ошибок и по-разному интерпретируют параметры. Старшие версии браузеров отображают рамки вокруг элементов более корректно.

|  |  |  |  |
| --- | --- | --- | --- |
| **Свойство** | **Значение** | **Описание** | **Пример** |
| **padding-top**  **padding-right**  **paddingbottom**  **padding-left**  **padding** | Значение  % | Отступ от границы элемента до его содержимого | table {padding:15px 15px} |
| **border-topwidth**  **border-right-width**  **borderbottom-width**  **border-leftwidth**  **border-width** | Thin  Medium  Thick  Значение | Ширина границы | P {border-top-width:4px} |
| **border-color** | Цвет | Цвет границы | P {border-color:red} |
| **border-style** | None  Dotted  Dashed  Solid  Double  Groove  Ridge  Inset  Outset | Стиль рамки | table {border-style:double} |
| **border-top**  **border-right**  **border-bottom**  **border-left** | border-top-width  border-style  Цвет | Определяет толщину, стиль и цвет каждой границы | table {border-top:solid 4px red; border-left: solid 4px blue} |
| **border** | border-top-width  border-style  Цвет | Задает толщину,  стиль и цвет рамки | table {border: solid 4px red} |

**Типы рамок**

Для управления видом рамки предоставляется восемь значений параметра border-style. Результат их действия представлен ниже.



Пример 2. Создание рамки

<p style="color: yellow; background-color: deepskyblue; text-decoration: underline; text-transform: uppercase; border: pink inset 25; PADDING: 20; font-size: larger; line-height: 40px; text-align: center;"> ... </p>

Реализация примера 2



Атрибут style="…" задает стилевое оформление абзаца.

Атрибут color: yellow; задает цвет текста.

Атрибут background-color: deepskyblue; задает цвет фона для абзаца.

Атрибут text-decoration: underline; задает подчеркивание для текста.

Атрибут text-transform: uppercase; задает режим заглавных букв для текста.

Атрибут border: pink inset 25; задает рамку вокруг абзаца, соответственно, розовую выпуклую толщиной 25 пикселей.

Атрибут PADDING: 20; задает расстояние между текстом и внутренними границами рамки.

Атрибут font-size: larger; задает размер шрифта.

Атрибут line-height: 40px; задает межстрочный интервал.

Атрибут text-align: center; задает выравнивание текста внутри абзаца по центру.

**Задание к выполнению лабораторной работы**

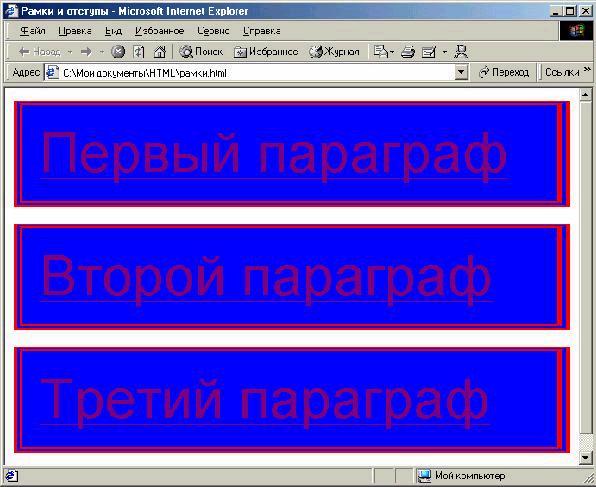
**1.1 Создание графического маркированного списка**

Создать в графическом редакторе Paint небольшой рисунок, который будет использоваться в качестве маркера списка. Создать маркированный список со своим маркером, используя стилевое свойство LIST-STYLE-IMAGE: url(имя\_рисунка).

Например, <UL style="LIST-STYLE-IMAGE: url(star2.jpg)">

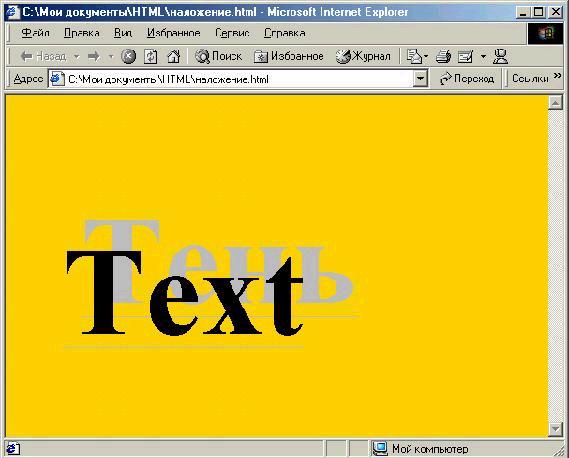
**1.2 Создание вложенных рамок**

Используя свойства блоков текста, создать следующую страницу:



**1.3 Создание наложенного текста друг на друга**

Значения отступов вокруг объектов можно указывать как положительные, так и отрицательные. Таким образом, можно использовать данную возможность для наложения одного слоя текста на другой. Например, можно создать текст с тенью, без использования трехмерного изображения. Создайте два стиля, которые отличаются цветом и размером отступов вокруг них. Используя отрицательные значения отступов и подбирая нужное значение, можно добиться того, что верхний слой как бы наползает на предыдущий. В результате у вас должно получиться следующее:



Для упрощения задачи используйте контейнер <DIV> и свойства margin-top и margin-left.

**Контрольные вопросы**

1. В чём может измеряться размер шрифта?

2. Как можно задать толщину рамки и цвет для неё?

3. Какие типы рамок Вы знаете?

4. Можно ли задать для сторон рамки различные цвета, толщину и т.д.?

**Литература**

1. Матросов А.В., Сергеев А.О., Чаунин Н.П. HTML 4.0 – СПб.:БХВ – Петербург, 2002. – 672с.:ил.
2. Пауэлл Томас А. WEB – дизайн: Пер. с англ. СПб.:БХВ – Петербург, 2002. – 1024с.:ил.
3. Имери, Винс. Как сделать бизнес в Internet: Пер. с англ. – 3-е изд. – К.; М.; СПб.: Диалектика, 1998ю – 464с.:ил.
4. Успенский И.В. Интернет как инструмент маркетинга. - СПб.:БХВ – Петербург, 2000. – 256с.:ил.
5. А.Д. Александровский. Создание Web-страниц с использованием FrontPage98 и JavaScript. – М.: ДМК, 1998 – 368 с.:ил.
6. Крупник А. Поиск в Интернете: самоучитель (знакомство, работа, развлечение). – СПб.: Питер, 2002. – 272 с.:ил.
7. Хоффман П. Internet. К.: Диалектика, 1995. – 160 с.:ил.
8. Палмер С. VBScript и Active: библиотека программиста. – СПб.: ЗАО «Издательство «Питер», 1999, -368 с.:ил.
9. Монкир, Майкл. Освой самостоятельно JavaScript за 24 часа.: Пер. с англ.: Уч.пос. – М.: «Издательский дом «Вильямс», 2000, -304 с.:ил.
10. И. Мавджой. ASP. Web – профессионалам: Пер. с англ. – К.: Издательская группа BHV, 2001. – 288с.