МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра компьютерных технологий

«Допущен к защите»
Заведующий кафедрой _____
_____
(Ф.И.О., ученая степень, звание)
_____ «_____»_____ **20__ г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка сайта научно-исследовательского центра „Империя пространства и безопасность"

Специальность Вычислительная техника и программное обеспечение

Выполнил (а) Жунусова И.М. ВТу-13-1
(Фамилия и инициалы) группа

Научный руководитель Аманбаев А.А. к.ф-м.н., доцент
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Бекмуева А.И.
(Фамилия и инициалы, ученая степень, звание)
_____ «19» 05 20 16 г.
(подпись)

по безопасности жизнедеятельности:
Мазалов И.Ф., канд. проф.
(Фамилия и инициалы, ученая степень, звание)
_____ «17» 05 20 16 г.
(подпись)

по применению вычислительной техники:
Аманбаев А.А. к.ф-м.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «06» 06 20 16 г.
(подпись)

Нормоконтролер: Аманбаев А.А. к.ф-м.н., доцент.
(Фамилия и инициалы, ученая степень, звание)
_____ «_____»_____ 20___ г.
(подпись)

Рецензент: Зауербек Нургали Сабырович д.т.н., доц
(Фамилия и инициалы, ученая степень, звание)
_____ «03» 06 20 16 г.
(подпись)

Алматы 2016 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Архитектурных и информационных технологий
Специальность Вычислительная техника и программное обеспечение
Кафедра Компьютерных технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Жунусова Нургуль Мамарбековна
(фамилия, имя, отчество)

Тема проекта Разработка сайта научно-исследовательского центра "Интернет пространство и безопасность"

утверждена приказом ректора **№149 от «23» октября 2015 г.**

Срок сдачи законченной работы « ___ » _____ 20___ г.

Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта
разработка сайта научно-исследовательского центра "Интернет пространство и безопасность", диаграмма.

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:
1) Разработка структуры сайта;
2) Выбор программного инструментария;
3) разработка удобного и понятного пользовательского интерфейса;
4) разработка оптимальной навигационной структуры.

Перечень графического материала (с точным указанием обязательных чертежей)

С точным указанием обязательных чертежей презентация в программе PowerPoint.

Рекомендуемая основная литература

Ермолаева З.Р., Бекенеева А.И. Методические указания к выполнению экономической части дипломных работ, 2013 г.

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
БЖД	Мазаков И.Ф.	11.03.-17.05	
Экон. часть	Бекенеева А.И.	03.03-19.05	
Нормоконтроль	Аманбаев А.А.	30.05-06.06.	
Программная часть	Аманов А.А.	20.05-06.06	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1)	Современное состояние разработки сайтов	1.03 - 20.03.16г	
2)	Разработка информационной системы сайта	15.03 - 5.05.16г	
3)	Разработка сайта НЧУ "Интегрия пространство и безопасность"	15.03 - 16.05.16г	
4)	Экономический анализ	08.03 - 19.05.16г	
5)	Безопасность жизнедеятельности	11.03 16.05. 16	

Дата выдачи задания «____»_____20___ г.

Заведующий кафедрой _____ _____
(подпись) (Фамилия и инициалы)

Руководитель _____ Ахмадеев А.А.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Жумарова А.М.
(подпись) (Фамилия и инициалы)

АННОТАЦИЯ

В предложенной дипломной работе представлено создание сайта научно-исследовательского центра «Интернет пространство и безопасность».

Целью работы является создание сайта, освещающего работу научно-исследовательского центра «Интернет пространство и безопасность». На сайте предоставлен разнообразный материал, раскрывающий работу научно исследовательского центра, а также все вопросы связанные с проводимыми научными исследованиям. Сайт логически поделен на две части – пользовательская часть и часть администратора. В процессе разработки сайта были использованы среды разработки – серверный язык программирования PHP, Apache и язык разметки гипертекста HTML. Также рассмотрены их особенности и возможности.

В пояснительной записке описаны этапы проектирования сайта, приведены требования к программному обеспечению, функциональное назначение сайта, приведено описание логической структуры, показаны скриншоты работы и сайта.

В приложении представлен листинг сайта.

ANNOTATION

Creation of a site of the research center “Internet Spase and Safety” is presented in the offered thesis.

The purpose of work is creation of the site shining work of the research center “Internet Spase and Safety”. On a site the versatile material opening work of scientifically research center, and also all questions connected with carried out scientific is provided to researches. The site is logically divided into two parts – the user part and part of the administrator. In the course of development of a site development environments – the server RNR, Apache programming language and a markup language of the hypertext of HTML were used. Their features and opportunity are also considered.

In the explanatory note site design are described, requirements to the software, a functional purpose of a site provided, the description of logical structure is provided, screenshots of work of a site are shown.

Site listing is presented in the appendix.

АҢДАТПА

Ұсынылған дипломдық жұмыста «Интернет кеңістік және қауіпсіздік» ғылыми-зерттеу орталығының сайтын құру келтірілген.

Жұмыстың мақсаты – «Интернет кеңістік және қауіпсіздік» ғылыми зерттеу орталығының жұмысымен таныстыратын сайт құру болып табылады. Сайтта ғылыми-зерттеу орталығының жұмысы туралы жан-жақты материал келтірілген. Сонымен қатар, өткізілетін ғылыми зерттеулерге байланысты сұрақтар көрсетілген. Сайт екі логикалық бөлімнен тұрады – қолданушы бөлімі мен әкімшілік бөлімі. Сайтты құру кезінде PHP серверлік бағдарламалау тілі, Apache және гипермәтінді белгілеу тілі – HTML қолданылған. Олардың ерекшеліктері мен мүмкіндіктері қарастырылған.

Түсініктеме жазбада сайты жазбада сайты жобалау кезеңдері суреттелген, бағдарламалық құралдарға қойылатын талаптар, сайттың функционалдық тағайындалуы келтірілген, логикалық құрылымы мен сайттың жұмыс жасау скриншоттары көрсетілген.

Қосымшада сайт листингісі келтірілген.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В последние годы в Казахстане быстрыми темпами развивается Интернет, и большая доля населения имеет у себя дома или на работе выход к его ресурсам, возникает необходимость использования этого направления в рекламных целях.

В вопросе разработки и создания Web-страниц в сети Интернет накоплен огромный багаж различных методов, способов и технологий, многие из которых, к сожалению, сейчас уже являются условно применимыми. Дело в том, что обновление аппаратного оборудования год от года только прогрессирует, причем с нарастающими темпами (до сих пор еще действуют эмпирические законы американского специалиста в области компьютерной техники Мура – каждые полтора года мощность микропроцессоров удваивается, и каждый год стоимость компонентов персонального компьютера снижается вдвое). Поэтому представляется актуальным и практически важным рассмотреть проблему проектирования Web-сайта в современных условиях с использованием всего спектра достижений, накопленных в данной области.

Наличие собственного веб-сайта сегодня считается не просто делом поднятия престижа, но и привлечение пользователей, но и просто необходимостью. «Если Вы не представлены в Интернете – Вы просто не существуете » - этой фразой можно описать значение веб-сайта.

В сети Интернет при упоминании в статьях или новостях ссылки делаются на корпоративный сайт (веб узел). На сегодняшний день уже не встает вопрос: нужен сайт или не нужен, но очень актуальна тема, как сделать сайт максимально эффективным с точки зрения успешной коммуникации и поддержания благоприятного имиджа. Поэтому создание и поддержка (регулярное обновление информации) сайта являются одним из важных компонентов деятельности в сети Интернет. World Wide Web (WWW) – глобальная компьютерная сеть на сегодняшний день содержит миллионы сайтов, на которых размещена всевозможная информация. Люди получают доступ к этой информации посредством использования технологии Internet. Для навигации WWW используются специальные программы Web-браузеры, которые существенно облегчают путешествие по бескрайним просторам WWW. Вся информация в Web-браузере отображается в виде Web-страниц, которые являются основным элементом байтов WWW.

Каждый из нас уже сейчас может сделать свой вклад в развитие Internet. Для этого достаточно создать свой веб-сайт и разместить его в Сети. О своевременности и актуальности рассматриваемой проблемы говорит тот факт, что большую часть своего времени программисты тратят на создание и внедрение сайтов в сеть интернет.

1. Теоретические основы проектирования и создания Web-узла

1.1 Основные понятия и определения

Веб-сайт – это совокупность Веб-страниц с повторяющимся дизайном, объединенных по смыслу, навигационное и физически находящийся на одном Веб-сервере. Веб-страница – самостоятельная часть Веб-сайта; документ, снабженный уникальным адресом (URL). Веб-страница может иметь статическое или динамическое построение. Обычно Веб-страницы организуются в виде гипертекста с включениями текста, графики, звука, видео или анимацию.

В сети Интернет просмотр Веб-страниц осуществляется по средствам браузера Разработка Web-узлов – это сложный процесс, включающий планирование узла, создание понятного интерфейса пользователя, подбор команды разработчиков, использование соответствующих технологий и общение с клиентом. Web-узлы, или, как их называют, сайты, представляют собой систему связанных между собой страниц, сходных по теме, оформлению, назначению и близкому (по меркам Интернета) расположению. Тематика, сложность и размеры сайтов могут значительно различаться. Под это определение попадают, как узлы университетов, фирм, компаний разработчиков, предлагающих и рекламирующих продукцию или услуг той или иной организации, так и виртуальные гиперсистемы, состоящие из тысяч разнородных документов и мультимедийных файлов, связанных многочисленными перекрестными ссылками. При создании и наполнении сайта всегда нужно помнить два принципа: уникальность и достоверность публикуемых материалов. Уникальность является первоочередным требованием к содержанию. В WWW уже может существовать немало страниц с похожими материалами. Web-узел должен чем-то отличаться от серверов с аналогичной тематикой, хотя бы для того, чтобы привлечь к себе внимание.

Наличие уникальных материалов на вышей странице увеличит ее посещаемость. Для того чтобы создать уникальный информационный ресурс, не обязательно изобретать что-то принципиально новое, может по-другому оформить уже существующие ресурсы, но при этом не заставлять клиента тратить много времени на их поиски. Проверить же ресурсы на уникальность можно с помощью серверов. Что касается авторитетности, то все зависит от того, насколько тщательно вы подберете информацию, проверите, ее и будете своевременно обновлять. При создании сайта необходимо помнить, что составляющие его отдельные документы должны быть объединены общим стилем оформления и средствами навигации. Единый стиль оформления – один из показателей, отличающих профессиональный Web-узел от любительского. Благодаря единообразно сделанным документам пользователи будут отличать ваш Web-узел от

других и запомнят его. Это не значит, что документы должны быть похожи друг на друга как две капли воды, но общая идея, единый стиль, должны присутствовать непременно. То же относится и к средствам навигации по страницам. Не стоит рассчитывать, что посетитель знает структуру сайта так же хорошо, как вы. Он должен без труда понимать, где он находиться сейчас и как можно попасть в любое другое место. Необходимо предусмотреть возможность перехода к первому документу, программе поиска к схеме Web-узла. Кроме того, единство стиля позволяет использовать шаблоны – страницы, содержащие только общие элементы оформления и навигации (без информационного накопления). С их помощью можно быстро и эффективно создавать новые страницы и распределять работу по их созданию между несколькими людьми. При использовании шаблона для получения готовой страницы достаточно лишь внести в него необходимую информацию. Последовательность, логичность, постоянство – вот необходимые качества хорошего Web-узла. Значительно упростят работу по оформлению и изменению стиля вашего сайта каскадные таблицы стилей. После того, как определены цели, задана структура и собрана текстовая и графическая информация, необходимо разработать внешний вид Web-узла. Он также зависит от тех целей, которые необходимо достичь. Спектр возможных решений здесь очень широк: от просмотра уже существующих страниц и создания подобных до обращения за помощью к профессиональным дизайнерам и художникам. В то же время, необходимо помнить о некоторых уже сложившихся правилах построения Web-документов, из которых состоит Web-узел.

Структура. На сегодня представления о структуре документа достаточно устоялось. Web-документ должен содержать в себе следующие разделы: заглавие, название, навигационную панель, собственно содержание, контактную информацию, дату и время обновления, авторские права и статус документа.

Логотип. Создавая Web-страницу, необходимо позаботиться о том, чтобы название всегда присутствовала на экране. Для этого в начале каждого Web-документа обычно помещается красочно оформленный логотип. Кроме того, название должно присутствовать и в выходных ко всем документам.

Навигационная панель. Одним из наиболее важных разделов Web-документа является навигационная панель или панель управления. WWW завоевала весь мир во многом благодаря тому, что гипертекстовые ссылки обеспечивают полную связность публикуемых материалов. Но эти же ссылки таят в себе опасность погружения в полный хаос, когда пройдя цепочку из трех–четырех документов, вы уже не сможете вернуться обратно, запутавшись в обилии ссылок. Ваш Web-узел должен обеспечить пользователю ясные и интуитивно понятные навигационные маршруты. Многочисленные исследования показали, что посетители Web-серверов

очень нетерпеливы и дальше, чем на два уровня документов, углубляться в содержание сервера не хотят.

Поэтому, создавая Web-узел большого объема, следует предусмотреть промежуточные документы, обычно находящиеся на первом-втором уровнях, от которых любая информация находится не далее, чем в двух переходах.

Принципы построения системы навигации:

- Навигация подразделяется на две части – глобальную и местную (реализованы, например, в виде верхнего и левого бокового меню).

- Глобальные «магистрали» должны позволять посетителю быстро перемещаться из одной точки в другую, а местные «дорожки» должны позволять перемещаться в пределах одного раздела, или группы документов.

- Все элементы навигации должны иметь ясные названия или условные обозначения.

- Сделайте ссылки на домашнюю страницу (нарисуйте домик, кнопку «домой» или что-нибудь оригинальное).

- Контекстные ссылки – встречаются внутри текста, служат для пояснения и являются дополнительным средством навигации по узлу. Не следует вставлять чрезмерное количество контекстных ссылок – это может отвлечь посетителя от этой страницы. Страницы Web взаимосвязаны. Переходы на другие страницы осуществляются с помощью выбора текста или рисунков, называемых ссылками на рисунок 1.

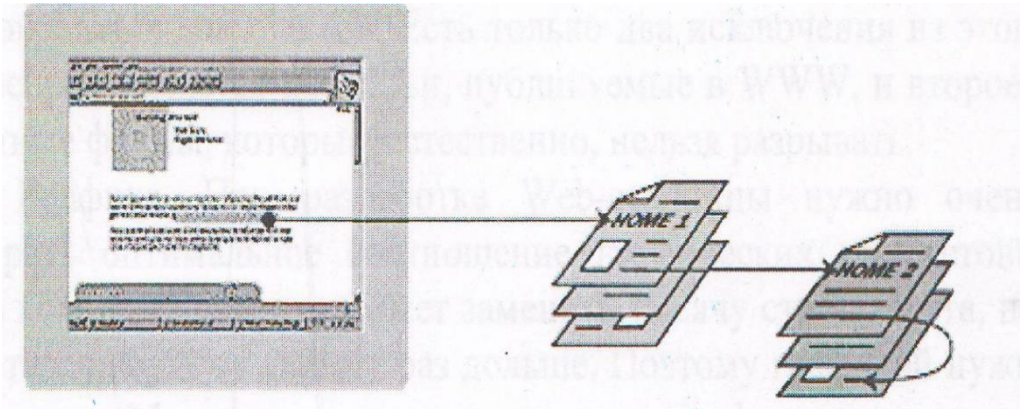

Рисунок 1. Организация перехода в виде ссылок

Ссылки являются подчеркнутыми или ограниченными словами и рисунками, которые содержат внедренные в них адреса Web (известные также как адрес URL – Universal Resource Locator). При выборе ссылки выполняется переход на определенную страницу конкретного узла. Отличить ссылку не составляет труда. Навигационная панель вашего сайта должна присутствовать в каждом документе. В первую очередь она долдна включать

в себя направляющие ссылки типа «Вперед»-«Назад» («Следующий»-«Предыдущий»), указывающие на соседние документы в стркутуре Web-узла. Далее от панели управления должны идти ссылки на все крупные разделы Сайта – так называемые разделы первого уровня. И, наконец, пользователь всегда должен иметь возможность мгновенно вернуться на главную страницу Web-узла. Помимо ссылок следует указать путь к локальной поисковой системе и индексу.

Содержание. Прежде всего, следует отметить, что содержание Web-документа должно в полной мере отвечать всем требованиям, предъявляемым к обычным газетам или журнальным публикациям: грамматическая и орфографическая корректность, точность и достоверность предлагаемых материалов и многое другое. Кроме того, появляется целый ряд специфических требований, которым должен удовлетворять Web-документ. Часто возникает вопрос о размерах документа: какое число страниц является оптимальным? Ответ на первый взгляд может показать странным: одна экранная страница или вообще никаких ограничений. Многочисленные исследования показали, что пользователи не любят работать с полосами прокрутки браузеров. Больше всего им нравятся документы, которые размещаются на одной экранной странице. Так в WWW – вы никоим образом не сможете дать пользователю больше информации, чем в концентрированном изложении на одной странице. Если все-таки вы не укладываетесь в эти рамки, создайте еще один документ. Одна экранная страница оказалась подходящей мерой представления информации. Если размер документа превышает одну страницу, то в большинстве случаев он может быть поделен на несколько логических частей, каждая их которых будет занимать не более одной страницы. Если же логического деления информации произвести не удается, то необходимо переработать стиль изложения, а может быть, и сами материалы. Сейчас выработалось единое мнение, что Web-сервер необходимо строить на основе одно экранных документов. Есть только два исключения из этого правила. Оно не распространяется на статьи, публикуемых в WWW, и второе исключение – анкетные формы, которые, естественно, нельзя разрывать.

Графика. При разработке Web-страницы нужно очень внимательно выбирать оптимальное соотношение графических и текстовых материалов. Одна хорошая картинка может заменить тысячу строк текста, но и загружаться по сети она будет в тысячу раз дольше. Поэтому графикой нужно пользоваться осторожно. Можно исходить из того, что графики на странице должно быть чуть меньше, чем хочется Web-мастеру. Пользователям может просто не хватить терпения, и они закроют документ еще до того, как он полностью загрузиться. Задержка отклика системы вызывает у пользователя раздражение. Время задержки возрастает в зависимости от времени суток, по разным оценкам до 15-60 секунд. Теперь

представьте, что у клиента только модем на 19200 бит/с. Большего на казахстанских телефонных линиях достичь очень тяжело. Тогда за минуту, то есть до того, как клиент потеряет терпение, можно передать только около 170 Кбайт данных. Следовательно, размер документа не должен превышать этого значения. Следует отметить, что обычно панель управления, логотип и название фирмы выполняются в виде графических элементов. После создания макета можно приступить к его реализации с помощью языка HTML и иных средств, предлагаемых современными технологиями WWW.

Размещение. Завершив создание Web-узла, необходимо разместить его в Internet. Здесь возможны два варианта: первый – использовать компьютер, который вместе с Web-сервером и Web-узлом находится в вашем офисе и подключается к Internet по выделенной или коммутируемой линии; второй – воспользоваться для размещения сайта услугами специальных организаций. Правильный выбор провайдера, предоставляющего доступ к Web-странице, позволит пользователям с максимальным удобством получать необходимую информацию. Кроме того, поддержка Web-серверов специальных возможностей значительно облегчит разработку Web-узла. На что следует обратить внимание при выборе провайдера, размещающего ваш Web-узел на своем сервере? Пропускная способность каналов. Чтобы вашим посетителям не пришлось слишком долго ждать загрузки страниц, провайдер должен обладать надежным высокоскоростным соединением порядка 1-2 Мбит в секунду.

Поддержка сервером провайдера SSI (Server Side Includes, вставка на стороне сервера). Использование SSI позволяет Web-серверу вставлять небольшие объемы динамических данных непосредственно в пересылаемый пользователю HTML- документ. Запрошенная HTML-страница «просматривается» в поисках элементов SSI. Обнаружив такой элемент, сервер вставляет требуемую динамическую информацию. С помощью SSI можно включать один файл в состав другого, исполнять CGI-сценарии и передавать другою информацию. Необходимо уточнить? Какие именно функции SSI поддерживаются на сервере провайдера. Поддержка сервором провайдера CGI-сценариев. CGI (Common Gateway Interface, общий шлюзовой интерфейс) – спецификация позволяющая Web-серверу выполнять произвольные прикладные программы. В результате работы таких программ (сценариев, или «скриптов») создают HTML-документ. С помощью CGI-сценариев могут приниматься данные от пользователя, они позволяют организовать диалог на Web-страницах, запросы к базам данных и т.д. Создать CGI-сценарий можно с помощью любого популярного языка программирования: Perl, Basic, C,C ++, Pascal и т.п. поддержка моментальной перекодировки. К сожалению, для русского языка в Internet при работе на разных платформах (Windows, Mac, Unix и т.д.) приняты различные кодировки. Чтобы пользователю было легко просматривать страницы, Web-

сервер провайдера должен уметь автоматически перекодировать документы в зависимости от поступившего запроса. В противном случае либо содержать вашего сайта для некоторых посетителей будет нечитаемым, либо придется обеспечивать несколько копий Сайта – по одной на каждую поддерживаемую кодировку. Способ обновления страниц. Обычно страницы обновляются по протоколу FTP (File Transfer Protocol, протокол передачи файлов). Некоторые FTP-клиенты позволяют работать с файлами на компьютере провайдера так же, как с собственным диском, - копировать, удалять, переименовывать и т.п. как правило, возможность размещения сайта провайдер предоставляет своим пользователям за небольшую плату или бесплатно. Существуют службы, которые предоставляют место под Web-узлы бесплатно вместе с адресом электронной почты и другими услугами. Как правило, условием такого «бесплатного» размещения является выделение на ваших страницах некоторого места под рекламу. Кроме того, накладываются ограничения на размер ваших файлов.

1.2 Создание узла интернета

Создание узла Интернета или интрасети может быть разделено на две основные задачи. Первая – создание и накопление. Сюда входит разработка и оформление материалов и ресурсов, которые человек (группа, компания, правительство какой-либо страны) хочет сделать доступными для какой-либо аудитории. Для преобразования подобной информации используется простой язык программирования. Вторая задача – публикация этих ресурсов в сети. Заполнение сервера Web состоит из трех основных частей: проектирования узла, программирование и размещения его содержимого на сервере. Вначале необходимо рассмотреть формат и структуру узла (на который надо поместить информацию) и определить наилучший способ предоставления информации. Составив общий план, надо решить, как привести информацию к форме, доступной программам просмотра Web. Язык программирования, используемый для этого, называется HTML (Hypertext Markup Language). И, наконец, надо выбрать оптимальный способ размещения информации на сервере. По существу, узел Web состоит из одной или более страниц Web и одной или более домашних страниц. Страницы Web – это организованное собрание материалов узла Web; они могут быть связаны друг с другом. Домашняя страница – это вводная или главная страница узла. Во многих отношения страница Web напоминает печатную страницу. Это – набор слов, фраз, графики и других элементов, организованных в виде последовательного потока данных от начала страницы до ее конца. Самое большое различие страниц в том, что длина страницы Web не имеет физических ограничений. Кроме того, страницы Web содержат связи, соединяющие их между собой. Это означает, что их не надо просматривать в некотором определенном порядке или последовательности, как страницы большинства книг. Домашняя страница служит точкой входа в узел Web.

Обычно это вводная страница, содержащая приветствие и ссылки на страницы второго уровня, доступные на данном узле. Домашние страницы формируют первое впечатление, которое производит узел на пользователя. Некоторые маленькие узлы Web типа персональных страниц состоят только из домашней страницы, тогда как на больших узлах может быть несколько домашних страниц.

Узел Web. Организация материалов на узле WWW в чем-то похожа на написание книги или статьи. Вначале необходимо продумать структуру и формат, а также ответить на множество вопросов. О чем следует сказать в первую очередь? Как говорить об этом? Какие темы логически связаны или дополняют друг друга? Каков порядок изложения? Нужны ли рисунки или иная графика? В зависимости от объема и вида информации реализация этих вопросов может отказаться простой, а может и весьма трудной.

Организация узла. Организация страницы (Home Page) является одним из основных элементов навигации. Существует два основных вида страниц: презентационная и информационная. Презентационная страница создается для небольших сайтов, красиво оформляется, часто почти целиком состоит из графики и имеет малое количество ссылок. Такая страница, как правило, должна помещаться на один экран. Информационная страница создается с использованием минимума графики и содержит большое количество информации. Ее задача – продемонстрировать посетителю обилие информации, имеющееся на узле, или предоставить всю наиболее актуальную информацию.

Правила организации веб страницы:
- самая актуальная информация должна быть видна на одном экране;
-элементы навигации (меню) должны быть очевидны и заметны;
-желательно, чтобы элементы навигации помещались на один экран;
-веб страница должна ясно говорить, чему посвящен этот узел;
-предельный размер страницы – 70-90 килобайт, минимизируйте количество графики.

Диапазон способов организации узлов Web весьма широк: от узлов, имеющих строгую линейную структуру, до узлов, у которых нет четкой структуры. Обычно страницы располагают в иерархической или линейном порядке, а также в виде паутины. Вместо того, чтобы с самого начала жестко структурировать информацию надо убедиться, что связи между страницами – логичные, что они ведут пользователя к необходимой ему информации. Вероятно, для этого лучше всего подойдет некоторая комбинация различных способов организации информации.

Иерархическая организация. Узлы Web, которые следуют иерархической или древовидной организации, имеют единственную точку

входа в узле; остальные страницы располагаются на исходящих из нее ответвлениях. Данный подход удобен, если можно легко разбить свою информацию на категории и подкатегории. При иерархической организации узла Web к странице самого нижнего уровня ведет один и только один путь. Такая строгая структура узла может вызвать проблемы у пользователей. Например, если пользователь прошел га несколько уровней вниз по одному из путей, а потом решил попасть в другую часть дерева, то ему придется возвращаться обратно.

Линейная организация. Чтобы пользователи узла Webмогли читать содержание узла как книгу или журнал или чтобы они пришли по заданному пути от начала и до конца узла – необходимо выбрать линейную организацию. Какая-то может иметь несколько связей с примечаниями и дополнениями, но для продвижения дальше пользователь должен вернуться на нее снова. Продвижение по документу осуществляется кнопкой Next, а возврат к началу узла-кнопкой Privy.

Хорош тот узел Web, который интересно и приятно посещать. Он предоставляет всю информацию. Необходимую пользователю, и по нему легко перемещаться. Для создания хорошо смотрящихся Web-страничек можно смело использовать яркую графику, изящные шрифты и другие способы. Но, в то же время, необходимо выдерживать небольшой размер страницы для их быстрой загрузки на компьютеры пользователей. Некоторые соображения, полезные при разработке страницы Web, приведены ниже.

Страница должна привлекать внимание. Первые впечатления по-настоящему важны. Самая интересная информация не обеспечит значительного трафика, если страница узла не привлечет внимания пользователя и не вызовет у него желание посмотреть другие страницы узла.

Размер страницы должен быть небольшим. Чтобы отдельную страницу было приятно читать, следует ограничить ее содержимое одной темой. Пользователь быстро устанет и разочаруется, если при поиске конкретной информации ему придется отфильтровывать большой объем лишних сведений. Лучше создавать маленькие страницы. Эта мера дает еще и выигрыш во времени просмотра и загрузки Web-страниц. А если имеющийся материал не умещается на одной странице нормального размера, то можно разделить его на несколько страниц и обеспечить связи между нами.

1.3 Структура HTML документа

Поскольку HTML-документы записываются в ASII-формате, то для ее создания может быть использован любой текстовый редактор. Обычно HTML-документ – это файл с расширением. htm, в котором текст размечен HTML-тегами (англ. Tag – специальные встроенные указания). Средствами

HTML задаются синтаксис и размещение тегов, в соответствии с которыми браузер отображает содержимое Веб-документа. Текст самих тегов Веб-браузером не отображается. Все теги начинаются символом <. Обычно имеется пара тегов – стартовый (открывающий) и завершающих (закрывающий) тег (похоже на открывающиеся и закрывающиеся скобки в математике), между которыми помещается размечаемая информация. Завершающий тег отличается от стартового лишь тем, что у него перед текстом в скобках стоит символ ‘/’ (слеш). Браузер, читающий HTML-документ, отображает его в окне, используя структуру HTML-тегов. В каждом HTML-документе должны присутствовать объявление HTML, заголовочная часть (HEAD) и тело документа (BODY). <HTML>-Сообщает браузеру, что документ создан на HTML. <HEAD>-Отмечает вводную и заголовочную части HTML-документа. <BODY>-Отмечает основной текст и информацию.

Эти теги необходимы Web-браузеру для определения различных частей HTML-документа, они не оказывают прямого влияния на внешний вид WEB-страницы. Они необходимы для того, чтобы последующие нововведения в HTML правильно интерпретировали страницу, а также для того, чтобы она выглядела одинаково в часто используемых браузерах. Например, на хостинге (место, где вы расположите веб-страницу) веб-сервером, создавая список имеющихся HTML-документом, запускается программа, которая использует только эти теги. Таким образом, если на странице нет этих тегов, она не будет включена в этот список. Список этот используется для поиска по хостингу. К примеру, если расположить страничку на Narod.RU, то при наличии этих тегов. Ваша страница будет доступна для поиска по Narod.RU.

<HTML> и </HTML>. Эти теги сообщают браузеру, что текст между ними следует интерпретировать как HTML-текст. Поскольку документы HTML чисто текстовые, тег <HTML> говорит о том, что файл написан на языке HTML (Hyper Text Markup Language – Язык гипертекстовой разметки).

Создавая новый HTML-файл, в первую очередь необходимо ввести данную пару тегов. Для того чтобы браузер принял этот текст необходимо набрать <HTML> в самом начале текста. Зачем набрать тег - </HTML> - в конце. Теперь весь текс, написанный между ними, будет принят браузерам за текст HTML. Во втором теге присутствует символ “/”. Правильный флэш (/) используется для обозначения закрывающих символов. Большинство HTML-тегов парные: один открывается (<HTML>), другой закрывается (</HTML>). Их действие распространяется только на тот текст, который находится между ними.

Объявление HTML <HTML> и </HTML>. Пара этих тегов сообщает программе просмотра (браузеру), что между ними заключен документ в формате HTML, причем первым тегом в документе должен быть тег (в самом начале документа), а последним-(в самом конце документа).

Заголовочная часть <HEAD> и </HEAD>. Между этими тегами располагается информация о документе (название, ключевые слова для поиска, описание и т.д.). Однако наиболее важным является название документа, которое мы видим в верхней строке окна браузера и в списках «Избранное (Book Mark)». Специальные программы – спай деры поисковых систем используют название документа для построения своих баз данных. Для того чтобы дать название своему HTML-документу текст помещается между тегами.

Тело документа. Третьей главой частью документа является его тело. Оно следует сразу за заголовком и находится между тегами <BODY> и </BODY>. Первый из них должен стоять после тега, а второй – перед тегом. Тело HTML-документа – это место, куда автор помещает информацию, отформатированную средствами HTML.

Формирование текста. В разделе BODY все символы табуляции и конца строк браузером игнорируются и никак не влияют на отображение страницы. Поэтому перевод строки в исходном тексте HTML-документа не приведет к началу новой строки в отображаемом обозревателем тексте при отсутствии специальных тегов. Это правило очень важно помнить и не забывать ставить разделяющие строки тега, иначе у текста не будет абзацев, и он станет нечитаемым.

Для начала новой строки используется тег
 (сокр. от англ. Break-прервать). Этот тег приводит к отображению браузером дальнейшего текста сначала следующие строки. Закрывающий для него тег не используется. Он удобен, если требуется с какого-то места писать с новой строки без начала нового абзаца, например, в новом предложении. Повторное его использование позволяет вставить одну или несколько пустых строк, отодвинув следующий фрагмент страницы вниз.

Сплошной текст без промежутков читается не очень легко, его неудобно просматривать и находить нужные места. Разбитый на абзацы, текст воспринимается гораздо быстрее. Для начала нового абзаца используется тег <P> (англ. Paragraph-абзац). Этот тег, кроме начала новой строки, вставляет одну пустую строку. Но многократное повторение <P>, в отличие от
, не приведет к появлению нескольких пустых строк, останется все та же одна пустая строка.

Внутри скобок тега кроме его название могут размещаться также атрибут <A> (англ. Attributes - атрибуты). Они отделяются от названия и между собой пробелами (одним или несколькими), а пишутся в виде имя _ атрибута = “значение”. Если значение не содержит пробелов, то кавычки могут быть опущены, но так делать не рекомендуется. Тег может содержать атрибут ALIGN, определяющий выравнивание абзаца. По умолчанию абзац выровнен влево ALIGN= “left”. Возможны также выравнивания вправо

ALIGN= “right” и по центру ALIGN= “center”. При использовании атрибутов, после форматируемого текста следует использовать закрывающий тег </A>.

В HTML-документе, кроме текста, могут содержаться горизонтальные разделительные линии. Они, как и текст, не требуют никаких внешних файлов. Тег <HR> выведен горизонтальную линию единичной толщины вдоль всей ширины страницы. Горизонтальная разделительная линия всегда приводит к разрыву строки, но пустых строк между линией и текстом не появляется. Тег <HR> может содержать несколько атрибутов и дают контурную линию с трехмерным эффектом углублению дает сплошную черную линию. Линия может не простираться во всю ширину страницы, а составлять лишь некоторую часть. Атрибут WIDTH задает ширину линии, в процентах от ширины всей страницы или в пикселях. Например, 50% - половина ширины, 400-ширина в 400 пикселов.

1.4 Постановка задачи

Как было отмечено выше, количество прикладных приложений в Интернете стремительно увеличивается и особенно актуально проекты, позволяющие решать многие задачи.

С учетом большого количества ресурсов располагаемых в интернет, современные Web-узлы как правило позволяют сетевые СУБД. Применение таких СУБД позволяет быстро и эффективно обрабатывать запросы пользователя и надежно сохранять данные. В дипломной работе будет разработан Web-узел, посвященный работе научного-исследовательского центра «Интернет пространство и безопасность».

Для этого необходимо решить следующие частные задачи:

-ознакомиться с современными технологиями и использовать их в своей разработке;

-изучить программыный инструментарий, применяемый для разработки и создания Web-сайтов;

-ознакомиться с основными правилами и рекомендациями по разработке и созданию Web-сайтов и неукоснительно им в своей практике;

-определяться со структурой Web-страниц;

-выбрать стратегию разработки и создания Web-сайта.

К конкретным задачам относиться:

-разработка структуры Web-узла;

-выбор программного инструментария;

-разработка удобного и понятного пользовательского интерфейса;

-разработка структуры и форм запросов;

-разработка оптимальной навигационной структуры системы;

-разработанный Web-проект должен быть оптимизировать под использование в реальной сети (иметь минимально возможный объем запрашиваемых Web-страниц).

Система должна быть ориентирована на дальнейшее развитие.

2. Проектирование и создание сайта научно исследовательского центра «Интернет пространство и безопасность»

В данной дипломной работе решается вопрос о создании и размещении в информационном пространстве WWW (World Wide Web – Всемирная паутина) собственного сайта научно-исследовательского центра «Интернет пространство и безопасность».

При решении этого вопроса идет пошаговое решение этой проблемы, т.е.: стиль оформления, необходимый для создания и последующего функционирования затраты, формат представления информации для размещения в Web, инструментарий и требования, предъявляемые к программному обеспечению Web-сервера и каналам связи с Internet.

Проект ориентирован на освоение и внедрение в профессиональную практику новых сетевых Интернет-технологий, а именно – современных инструментальных средств разработки и создания перспективных Web-сайтов, с использованием традиционных технологий, базирующихся на использовании HTML-редакторов (уровня Note Pad и ему подобных) и выше.

2.1 Используемых технологии

В дипломной работе по созданию сайта НИЦ «Интернет пространство и безопасность» используются технология статических html-страниц, таблицы стилей CSS, язык программирования JavaScript, создание график Adobe Photoshop при использовании технология статических html-страниц, веб узел создавался в программе WordPress 2.0 и в обычном текстовом редакторе Блокнот (Note Pad).

Создается несколько страниц единым стилем и оформлением, один из которых называется Главной, а несколько других представляют собой содержание основных рубрик проекта. Затем все они связываются сетью гиперссылок, на них вывешивается информация, картинки и кнопки, выбирается фон.

Язык, который был взят за основу создания веб-узел (веб-страниц) для НИЦ и выполнения дипломной работы – это HTML (Hyper Text Markup Language). Термин HTML (Hyper Text Markup Language) означает «язык маркировки гипертекстов». Это понятие более широкое, оно включает в себя Интерне и локальные сети, редакторы, браузеры, разнообразные программные продукты, компакт-диски, обучающие курсы, дизайн и многое другое.

HTML. HTML – своеобразная противоположность сложным языкам программирования, известным только специалистам. Гипертекст подходит для включения элементов мультимедиа в традиционные документы. Практически именно благодаря развитию гипертекста, большинство

пользователей получило возможность создавать собственные мультимедийные продукты и распространять их на компакт-дисках. Такие информационные системы, выполненные в виде набора HTML-страниц, не требует разработки специальных программных средств, так как все необходимые инструменты для работы с данными (Web-браузеры) стали частью стандартного программного обеспечения большинства персональных компьютеров.

HTML как основа создания Web- страниц, имеет прямое отношение к созданию веб узла.

CSS. Как известно, CSS (Cascading Style Sheets) – это стандарт переопределения стилевых шаблонов для элементов HTML-страниц. С помощью различных средств CSS (селекторы классов и идентификаторов, псевдо классы и пр.) стандартный вид электронного документа можно легко превратить в привлекательную и стильную страницу. С помощью этой технологии был привлекательный вид веб узла для кафедры.

Скрипты (JavaScript). В данной дипломной работе были использованы такие скрипты как мерцание и блок вертикальных строк. Скрипт – программа, содержащая набор инструкций для некоторых приложений или утилит. Семантика и синтаксис инструкций в скриптах определяются соответствующими приложениями. Обычно язык скриптов включает простые структуры управления: линейные последовательности, циклы и условные выражения.

JavaScript – язык программирования, основанный на объектном представлении браузера. Текст программы встраивается непосредственно в HTML – документ и интерпретируется самим браузером. Скрипт-язык – в Интернет-интерпретируемый (объектное ориентированный) алгоритмический язык, предназначенный для генерации динамических и статических веб-страниц. JavaScript применялся для придания пользовательскому интерфейсу большей интерактивности по сравнению с обычными статическим HTML страницами без использования скрипт-язык, он позволял реализовать: бегущую строку, мерцание баннера и т.д. Пример использования JavaScript для баннера (мерцание):

```
</script>

<script language = “javascript” type = “text/javascript”>

<!-function MM_swapImgRestore() {

vari,x,a=document.MM_sr;for(i=0;a&&i<a.length&&(x=a[i]&&x.oSrc;i++)
x.src=x.oSrc; }
```

```
function MM_preloadImages() {

var d=document; if (d.images){ if(!d.MM_p) d.MM_p=new Array();

var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length; i++)

if(a[i].indexOf("#")!=0){d.MM_p[j]=newImage;d.MM_p[j++].src=a[i];}}
}
function MM_findObj(n,d) {
varp,i,x;if(!d)d=document; if((p=n.indexOf("?"))>0&&parent.frames.length)
{ d=parent.frames[n.substrihg(p+1)].document; n=n.substring(0,p);}

if(!(x=d[n]&&d.all) x=d/all[n]; for(i=0;!x&&i<d.forms.length;i++) x=d.forms[i][n];

for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].document);

if(!x&& d.getElementById) x=d.getElementById(n); return x;

}

function MM_swapImage() {

var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new Array; for(i=0; i<(a.length-2); i+=3)

if ((x=MM_findObj(a[i]))!=null) {document.MM_sr[j++]=x; if(!x.oSrc) x.oSr=x.src; x.src=a[i+2];}

} //-- > </script>
```

Adobe Photoshop – на сегодняшний день это самый мощный пакет для профессиональной обработки растровой графики. Это целый комплекс, обладающий многочисленными возможностями модификации растрового рисунка, имеющий огромный набор различных фильтров и эффектов, причем есть возможность подключать инструменты независимых производителей. (Пакет предлагает, например, средства для восстановления поврежденных изображений, ретуширования фотографий или создания самых фантастических коллажей, которые только может позволить себе наше воображение).

2.2 Спецификация HTML

Язык HTML приобрел популярность в середине 90-х годов, благодаря экспоненциальному росту сети Интернет. К этому времени назрела

необходимость стандартизации языка, поскольку различные компании, разрабатывавшие программное обеспечение для доступа в Интернет, предлагали свои варианты инструкции HTML, число которых все возрастало и возрастало. Настала пора прийти к какому-то единому соглашению в части применения тегов языка HTML.

Работу по созданию спецификации HTML взяла на себя организация, называемая World Wide Web Consortium (сокращенно – W3C). В ее задачу входило составление спецификации, отражающей современный уровень развития возможностей языка с учетом разнообразных предложений компаний разработчиков браузеров. Так, в ноябре 1995 г. появилась спецификация HTML 2.0, призванная формализовать сложившуюся к концу 1994 г. практику использования HTML.

Схема утверждения спецификаций состоит в следующем. Консорциум W3C выпускает проект спецификации, после обслуживания которого выпускается так называемый черновой, рабочий (draft) вариант спецификации и предлагает его к обслуживанию на определенный период. После периода обсуждения рабочий вариант спецификации может стать рекомендацией, т.е. официально признанным вариантом спецификации HTML.

Вскоре после спецификации 2.0 была выпущена рабочая версия спецификации 3.0, срок окончания периода обсуждения которой истек в сентябре 1995 г.

Эта спецификация не была принята в качестве официальной рекомендации. В нее планировали включить большое разнообразие тегов и возможностей, специфичных для отдельных браузеров, однако Консорциум W3C не нашел возможности разработать хорошую спецификацию для такого большого числа инструкций. После размышлений в мае 1996 г. был выпущен HTML 3.2. Проект основывался на части тегов, имеющихся в версии 3.0, которые показывали стабильность в работе. В сентябре 1996 г. после нескольких месяцев обсуждения версия 3.2 стала предлагаемой спецификацией, а в январе 1997 г. – официальной рекомендацией. Июль 1997 года ознаменовался выходной предлагаемой спецификацией HTML 4.0, которая в декабре 1997 г. стала официальной рекомендацией. На сегодняшний день это последняя из принятых спецификаций. В приводимом здесь кратком обзоре истории развития языка HTML вряд ли стоит детально описывать особенности различных спецификаций, тем более, что в реальной жизни разработчики далеко не всегда следуют рекомендациям Консорциума. Отметим лишь некоторые идеи, заложенные в основу последней спецификации. В спецификации HTML 4.0 ключевой идеей стало отделение описания структуры документа от описания его представления на экране монитора. Опыт показывает, что разделение структуры и преставления

документа уменьшает затраты на поддержку широкого спектра платформы, сред и т.п., а также облегчает внесение исправлений в документе. В соответствии с этой идеей следует шире пользоваться методами описания представления.

Официальной спецификации HTML 1.0 не существует. До 1995 года существовало множество неофициальных стандартов HTML. Чтобы стандартная версия отличалась от них, ей сразу присвоили второй номер.

Версия 3 была предложена Консорциумом всемирной паутины (W3C) в марте 1995 года и обеспечивала много новых возможностей, таких как создание таблиц, «обтекание» изображений текстом и отображение сложных математических формул, поддержка gif формата. Даже при том, что этот стандарт был совместим со второй версией, реализация его была сложна для браузеров того времени. Версия 3.1 официально никогда не предлагалась, и следующей версией стандарта HTML стала 3.2, в которой были опущены многие нововведения версии 3.0, но добавлены нестандартные элементы, поддерживаемые браузерами Netscape Navigator и Mosaic.

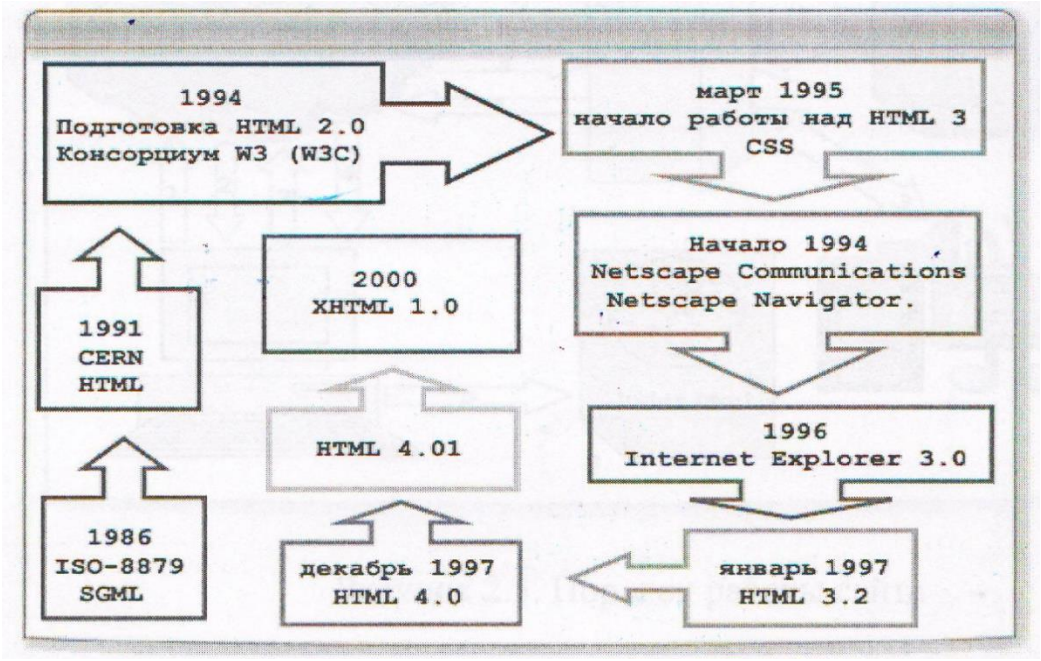

Рисунок 2.1. История развития HTML

Планируемая версия XHTML 2.0 должна была разорвать совместимость со старыми версиями HTML и XHTML, но 2 июля 2009 года консорциум всемирной паутины объявил, что полномочия рабочей группы XHTML2 истекают в конце 2009 года. Таким образом, была приостановлена вся дальнейшая разработка стандарта XHTML 2.0.

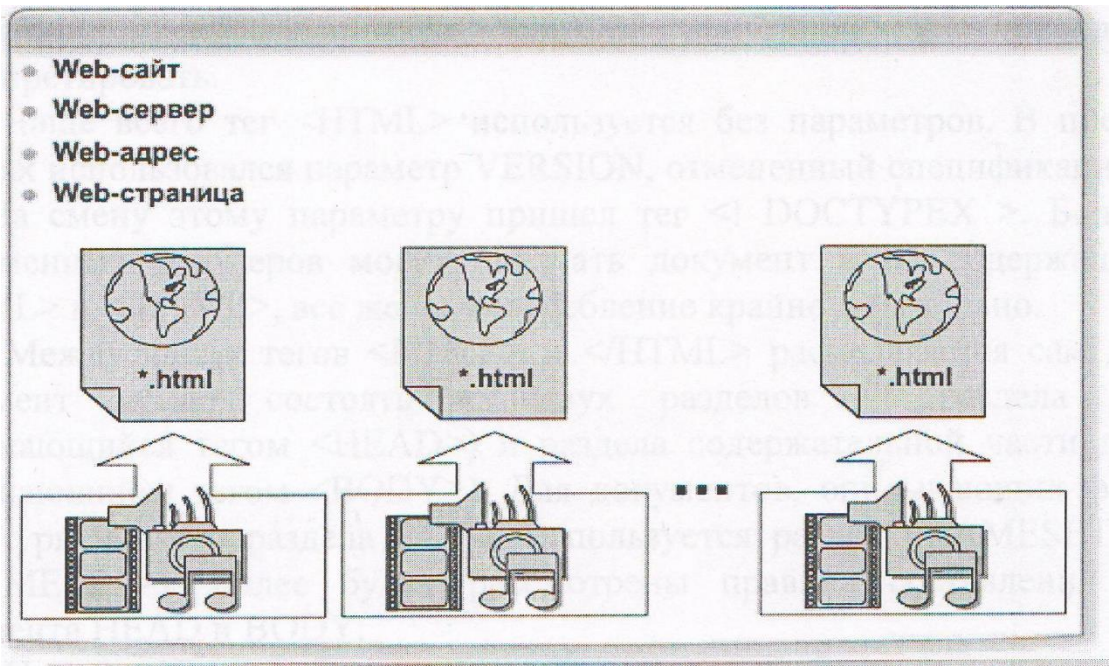

Рисунок 2.2. Структура Web-страниц

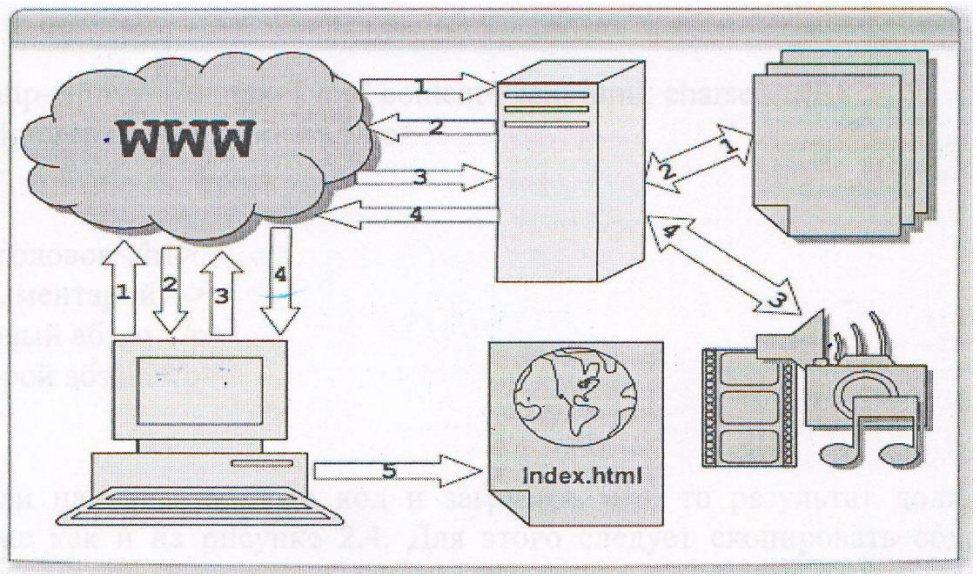

Рисунок 2.3. Порядок работы сайта

2.3 Структура документа

Первым тегом, с которого следует начинать описание документов HTML, а является тег <HTML>. Он должен всегда начинать описание документа, а завершать описание документа должен тег </HTML>. Эти теги обозначаются, что находящиеся между ними строками представляются единый HTML-документ. Сам по себе документ является обыкновенным текстом ASCII-файлом. Без этих тегов браузер или другая программа просмотра, возможно, будет не в состоянии идентифицировать формат документа и правильно его интерпретировать.

Чаще всего тег <HTML> используется без параметров. В предыдущих версиях использовался параметр VERSION, отмененный спецификацией HTML 4.0. На смену этому параметру пришел тег <! DOCTYPEX>. Большинство современных браузеров могут опознать документ и не содержащий тегов <HTML> и </HTML>, все же их употребление крайне желательно.

Между парой тегов <HTML> и </HTML> располагается сам документ. Документ может состоять из двух разделов – раздела заголовка (начинающийся тегов <HEAD>) и раздела содержательной части документа (начинающийся тегом <BODY>). Для документа, описывающих фреймовые структуры, вместо раздела BODY используется FRAMESET (с тегом < FRAMESET >). Далее будут рассмотрены правила составления разделов документа HEAD и BODY.

Пример. Исходный код веб-страницы:

```
<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4/01//EN”
http://www.w3.org/TR/html14/strict.dtd”>
<html>
<head>
<meta http-equiv = “Content-Type” content = “text/html; charset=utf-8”>

<title> Пример веб-страницы </title>

</head>

<body>

<h1> Заголовок </h1>

<!--Коментарий-- >

<p>Первый абзац.</p>

<p>Второй  абзац.</p>
```

<body>
<html>
</html>

Если написать данный код и загрузить его то, результат должен быть таким же как и рисунок 2.4. Для этого следует скопировать содержимое данного примера и сохранить его в папке c:/www/ под именем example41.html. После этого запустите браузер и откройте файл через пункт меню Файл . Откройте файл (Ctrl+O). В диалоговом окне выбора документа укажите файл example41.html. В браузере откроется веб-страница, показанная на рисунке 2.4.

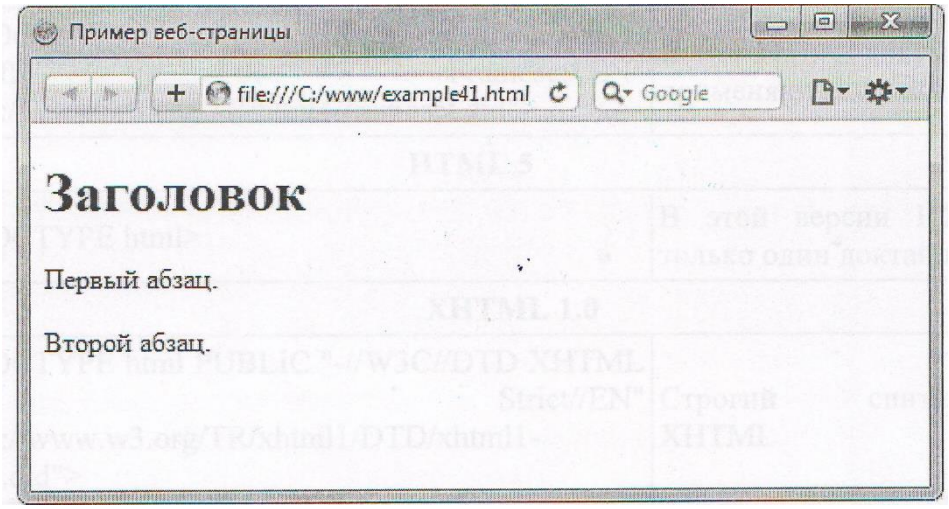

Рисунок 2.4. Результат выполнения примера

Далее разберем отдельные строки нашего кода.

<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4/01//EN”

http://www.w3.org/TR/html14/strict.dtd”>

Элемент <! DOCTYPE> предназначен для указания типа текущего документа – DTD (document type definition, описание типа документа). Это необходимо, чтобы браузер понимал, как следует интерпретировать текущую веб-страницу, ведь HTMLсуществует в нескольких версиях, кроме того, имеется XHTML (Extensible Hyper Text Marker Language, расширенный язык разметки гипертекста), похожий на HTML, но различающийся с ним по синтаксису. Чтобы браузер «не путался» и понимал, согласно какому стандарту отображать веб-страницу и необходимо в первой строке кода задавать <! DOCTYPE>. Существует несколько видов <! DOCTYPE>, они различаются в зависимости о версии HTML, на которую ориентированы. Разница между строгим и переходным описанием документа состоит в различном подход к описанию кода документа. Строгий HTML требует жесткого соблюдения спецификации HTML и не прощает ошибок.

Переходный HTML более «спокойно» относится к некоторым огрехам кода, поэтому этот тип в определенных случаях использовать предпочтительнее.

Например, в строгом HTML и XHTML непременно требуется наличие тега <title>, а в переходном HTML его можно опустить и не указывать. При этом помним, что браузер в любом случае покажет документ, независимо от того, соответствует он синтаксису или нет. Подобная проверка осуществляется при помощи валидатора и предназначена в первую очередь для разработчиков, чтобы отслеживать ошибки в документе.

В дальнейшем будем применять преимущественно строгий <! DOCTYPE>, кроме случаев, когда это оговаривается особо. Это позволит нам избегать типичных ошибок и приучит к написанию синтаксически правильного кода. Часто можно встретить код HTML вообще без использования <! DOCTYPE>, веб-страница в подобном случае все равно будет показана. Тем не менее, может получиться, что один и тот же документ отображается в браузере по разному при использовании <! DOCTYPE> и без него. Кроме того, браузеры могут по-своему показывать такие документы, в итоге страница «рассыпается» т.е. будет отображаться совсем не так, как это требуется разработчику. Чтобы не произошло подобных ситуаций, всегда добавляйте <! DOCTYPE> в начало документа.

<html>. Тег <html> определяет начало HTML-файла, внутри него хранится заголовок (<head>) и тело документа (<body>). <head>. Заголовок документа, как еще называют блок <head>, может содержать текст и теги, но содержимое этого раздела не показывается напрямую на странице, за исключением контейнера **<title>**. <meta http-equip = “Content-Type” content = “text/html; charset=utf-8”>. Тег <meta> является универсальным и добавляет целый класс возможностей, в частности, с помощью мета тегов, как обобщенно называют этот тег, можно изменять кодировку страницы, добавлять ключевые слова, описание документа и многое другое. Чтобы браузер понимал, что имеет дело с кодировкой UFT-8 (Unicodetransformationformat, формат преобразования Юникод) и добавляется данная строка. <title>Пример веб-страницы</title>. Тег <title> определяет заголовок веб-страницы, это один из важных элементов предназначенный для решения множества задач. В операционной системе Windows заголовок отображается в левом верхнем углу окна браузера на рисунке 2.5.

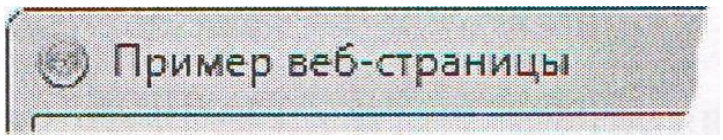

Рисунок 2.5. Вид заголовка в браузере

Тег <title> является обязательным и должен непременно присутствовать в коде документа. </head>. Обязательно следует добавлять

закрывающий тег </head>, чтобы показать, что блок заголовка документа завершен. <body>. Тело документа <body> предназначено для размещения тегов и содержательной части веб-страницы.

<h1> Заголовок</h1>. HTML предлагает несть текстовых заголовков разного уровня, которые показывают относительную важность секции, расположенной после заголовка. Так, тег <h1> представляет собой наиболее важный заголовок первого уровня, а тег <h6> служит для обозначения заголовка шестого уровня и является наименее значительным. По умолчанию, заголовок первого уровня отображается самым крупным шрифтом жирного начертания, заголовки последующего уровня по размеру меньше. Теги <h1> …<h6> относят к блочным элементам, они всегда начинаются с новой строки, а после них другие элементы отображаются на следующей строке. Кроме того, перед заголовком и после него добавляется пустое пространство.

<! --Комментарий -->. Некоторый текст можно спрятать от показа в браузере, сделав его комментарием. Хотя такой текст пользователь не увидит, он все равно будет передавать в документе, так что, посмотрев исходный код, можно обнаружить скрытые заметки. Комментарий нужен для внесения в код своих записей, не влияющих на вид страницы. Начинаются они тегом <! -- и заканчиваются -->. Все, что находится между этими тегами, отображаться на веб-странице не будет.

Первый абзац . Тег определяет абзац (параграф) текста. Если закрывающего тега нет, считается, что конец абзаца совпадает с началом следующего блочного элемента.

Второй абзац . Тег является блочным элементом, поэтому текст всегда начинается с новой строки, абзацы, идущие друг за другом разделяются между собой отбивкой (так называется пустое пространство между ними). Это хорошо видно на рисунке 2.6. </body>. Следует добавить закрывающий тег </body>, чтобы показать, что тело документа завершено. </html>. Последним элементом в коде всегда идет закрывающий тег </html>.

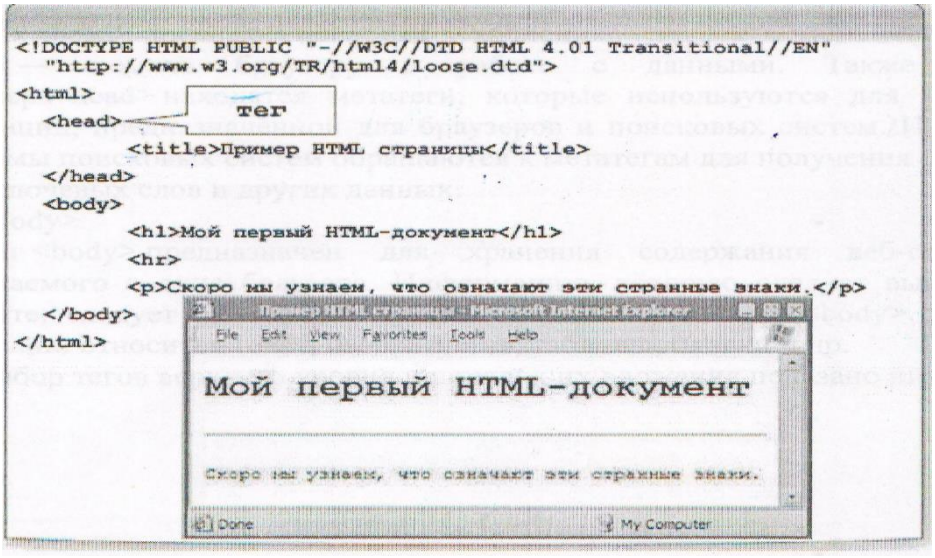

Рисунок 2.6. Пример простого сайта

Типы тегов. Каждый тег HTML принадлежит к определенной группе тегов, например, табличные теги направлены на формирование таблиц и не могут применяться для других целей.

Условно теги делятся на следующие типы: теги верхнего уровня; теги заголовка документа; блочные элементы; встроенные элементы; универсальные элементы; списки; таблицы.

Следует учитывать, что один и тот же тег может одновременно принадлежать разным группам, например, теги <ol> и <ul> относятся к категории списков, но также являются и блочными элементами.

Далее рассмотрим только те теги, которые потребуются нам в дальнейшей работе.

Теги верхнего уровня. <html>. Тег <html> является контейнером. Который заключает в себе всё содержимое веб-страницы, включая теги <head> и <body>. Открывающий и закрывающий теги <html> в документе необязательны, но хороший стиль диктует непременное их использование. <head>. Тег <head> предназначен для хранения других элементов, цель которых – помощь браузера в работе с данными. Также внутри контейнера <head> находятся мета теги, которые используются для хранения информации. Предназначенное для браузеров и поисковых систем. Например, механизмы поисковых систем обращаются к мета тегам для получения описания сайта, ключевых слов и других данных. <body>. Тег <body> предназначен для хранения содержания веб-страницы, отображаемого в окне браузера. Информацию, которую следует выводить в документе, следует располагать именно внутри контейнера <body>. К такой информации относится текст, изображения, таблицы, списки и др.

Набор тегов верхнего уровня и порядков их вложения показано ниже:

<Html>

<Head>…

</head>

<Body>…
</body>
</html>

В данном примере показано, что контейнер <html> определяет «каркас» всей веб-страницы, внутри него вначале задается тег <head>, затем идет контейнер <body>, в нем хранится содержательная часть документа, которая и отображается в браузере. Теги <html> и <body> хотя и не относятся к обязательным тегам (т.е. их можно не размещать в коде), все же стоит добавлять всегда. Это позволяет получить четкую и понятную структуру документа.

Заметьте, что в примере не упоминается <! DOCTYPE>, поскольку этот обязательный элемент кода веб-страницы не является тегом, а предназначен для браузеров, чтобы сообщить им, как интерпретировать текущий документ.

Теги заголовка документа. К этим тегам относятся элементы, которые располагаются в контейнере <head>. Все эти теги напрямую не отображаются в окне браузера, за исключением тега <title> который определяет название веб-страницы.

<title>. Используется для отображения строки текста в левом верхнем углу окна браузера, а также на вкладке. Такая строка сообщает пользователю название сайта и другую информацию, которую добавляет разработчик.

<meta>. Мета теги используются для хранения информации, предназначенной для браузеров и поисковых систем. Например, механизмы поисковых систем обращаются к мета тегам для получения описания сайта, ключевых слов и других данных. Хотя <meta> всего один, он имеет несколько атрибутов, поэтому к нему и применяется множественное число.

Так для краткого описания содержимого веб-страницы используется значение description атрибута name, как показано ниже:

<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01//EN”
“html://www.w3.org/TR/html4/strict.dtd”>
<Html>
<Head>
<Title>HTML</title>
<Meta name = “description” content = “Сайтов HTML и создании сайтов”>
<Meta http-equip = “content-type” content = “text/html; charset=utf-8”>
</head>
<Body>
…
</body>
</html>

Описание сайта, заданное с помощью тега <meta> и назначения description, обычно отображается в поисковых системах или каталогах при выводе результатов поиска. Значение keywords также предназначено в первую очередь для повышения рейтинга сайта в поисковых системах, в нем перечисляются ключевые слова, встречаемые на веб-странице которая указана ниже:

<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01//EN”
“html://www.w3.org/TR/html4/strict.dtd”>
<Html>
<Head>
<Title>HTML</title>
<Meta name = “keywords” content = “HTML, META, метатег, тег, поисковая система”>
<Meta http-equip = “content-type” content = “text/html; charset=utf-8”>
</head>
<Body>
…
</body>
</html>

Ключевые слова можно перечислять через пробел или запятую. Поисковые системы сами приведут запись к виду, который они используют.

Блочные элементы. Блочные элементы характеризуются тем, что занимают всю доступную ширину, высота элемента определяется его содержимым, и он всегда начинается с новой строки.

<block quote>. Предназначен для выделения длинных цитат внутри документа. Текст, обозначенный этим тегом, традиционно отображается как выровненный блок с отступами слева и справа (примерно по 40 пикселов), а также с пустым пространством сверху и снизу. . Тег относится к универсальным блочным контейнерам и применяется в тех случаях, где нужны блочные элементы без дополнительных свойств. Также с помощью тега можно выравнивать текст внутри этого контейнера с помощью атрибута align. <h1>,,<h6>. Эта группа тегов определяет текстовые заголовки разного уровня, которые показывают относительную важность секции, расположенной после заголовка. <hr.>. Рисует горизонтальную линию, которая по своему виду зависит от используемых атрибутов. Линия всегда начинается с новой строки, а после нее все элементы отображаются на следующей строке. . Определяет параграф (абзац) текста.

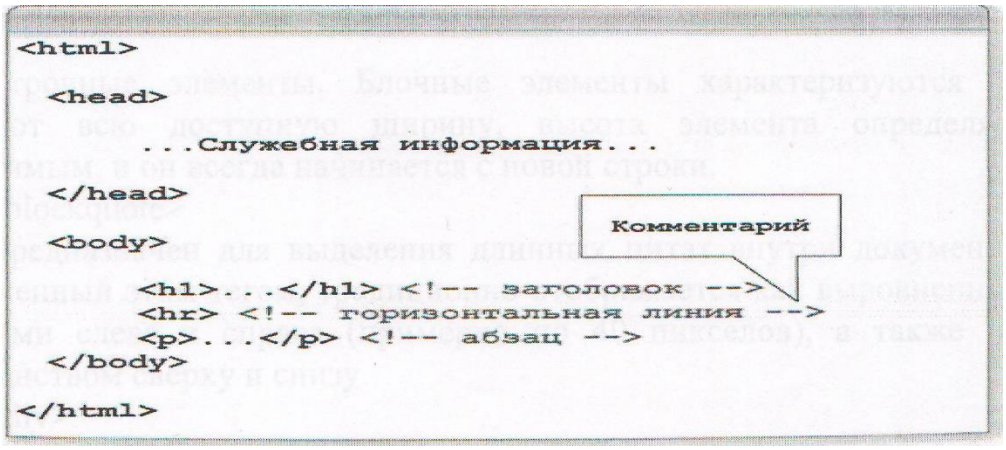

Рисунок 2.7. Тело веб-страницы

Задает блок предварительно форматированного текста. Такой текст отображается обычно моноширинным шрифтом и со всеми пробелами между словами. В HTML любое количество пробелов, идущих в коде подряд на веб-странице показывается как один. Тег <pre> позволяет обойти эту особенность и отображать текст как требуется разработчику.

Следующие теги не должны размещаться внутри контейнера <pre>:<big>, <img>, <small>, <sub> и <sup>.

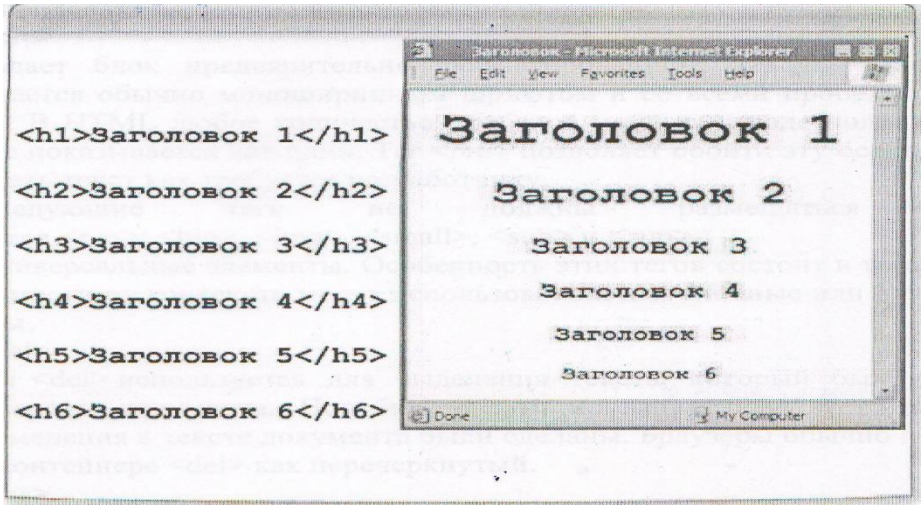

Рисунок 2.8. «Заголовок»

Строчные элементы. Блочные элементы характеризуются тем, что занимают всю доступную ширину, высота элемента определяется его содержимым, и он всегда начинается с новой строки.

<block quote>. Предназначен для выделения длинных цитат внутри документа. Текст, обозначенный этим тегом, традиционного отображается как выровненный блок с отступными слева и справа (примерно по 40 пикселов), а также с пустым пространством сверху и снизу.

. Тег относится к универсальным блочным контейнерам и применяется в тех случаях, где нужны блочные элементы без дополнительных свойств. Также с помощью тега можно выравнивать текст внутри этого контейнера с помощью атрибута align.

<h1>, <h6>. Эта группа тегов определяет текстовые заголовки разного уровня, которые показывают относительную важность секции, расположенной после заголовка.

<hr.>. Рисует горизонтальную линию, которая по своему виду зависит от используемых атрибутов. Линию всегда начинается с новой стрики, а после нее все элементы отображаются на следующей строке.

<p>. Определяет параграф (абзац) текста.

<pre>. Задает блок предварительно форматированного текста. Такой текст отображается обычно моноширинным шрифтом и со всеми пробелами между словами. В HTML любое количество пробелов в коде подряд на веб-странице показывается как один. Тег <pre> позволяет обойти эту особенность и отображать текст как требуется разработчику. Следующие теги не должны размещаться внутри контейнера <pre>:<big>, <img>, <small>, <sub> и <sup>.

Универсальные элементы. Особенности этих тегов состоит в том, что они в зависимости от контекста могут использоваться как блочные или встроенные элементы.

<del>. Тег <del> используется для выделения текста, который был удален в новой версии документа. Подобное форматирование позволяет отследить, какие изменения в тесте документа были сделаны. Браузеры обычно помечают текст в контейнере <del> как перечеркнутый.

<ins>. Тег <ins> предназначен для акцентирования вновь добавленного текста и обычно применяется наряду с тегом <del>. Браузеры помечают содержимое контейнера <ins> подчеркиванием текста.

Теги для списков. Списком называется взаимосвязанный набор отдельных фраз или предложений, которые начинаются с маркера или цифры. Списки предоставляют возможность упорядочить и систематизировать разные данные и представить их в наглядном и удобном для пользователя виде. . Тег <ol> устанавливает нумерованный список, т.е. каждый элемент списка начинается с числа или буквы и увеличивается нарастающей.

. Устанавливает маркированный список, каждый элемент которого начинается с небольшого символа-маркера.

. Тег <li> определяет отдельный элемент списка. Внешний тег <ui> и <ol> устанавливает тип списка-маркированный или нумерованный.

<dd>, <dt>, <dl>. Тройка элементов предназначена для создания списка определений. Каждый такой список начинается с контейнера <dl>, куда входит тег <dt> создающий термин и тег <dd> задающий определение этого термина. Закрывающий тег </dd> не обязателен, поскольку следующий тег сообщает о завершении предыдущего элемента. Тем не менее, хорошим стилем является закрывать все теги.

Теги таблиц. Таблицы состоит из строк и столбцов ячеек, которые могут содержать текст и рисунки. Обычно таблицы используются для упорядочения и представления табличных данных.

<table>. Служит контейнером для элементов, определяющих содержимое таблицы. Любая таблица состоит из строк и ячеек, которые задаются с помощью тегов <tr> и <td>.

<td>. Предназначен для создания одной ячейки таблицы. Тег <td> должен размещаться внутри контейнера <tr> , который в свою очередь располагается внутри тега <table>.

<th>. Тег <th> предназначен для создания одной ячейки таблицы, которая обозначается как заголовок. Текст в такой ячейке отображается браузером обычно жирным шрифтом и выравнивается по центру.

<tr>. Тег <tr> служит контейнером для создания строки таблицы.

Теги для фреймов. *Фреймы* разделяют окно браузера на отдельные области, расположенные вплотную друг к другу. В каждую из таких областей загружается самостоятельная веб-страница, определяемая с помощью тега **<frame>**. С помощью фреймов веб-страница делится на два или более документа, которые обычно содержат навигацию по сайту и его контент. Механизм фреймов позволяет открывать документ в одном фрейме, по ссылке, нажатой в совершенно другом фрейме. Допустимо также использовать вложенную структуру элементов, это позволяет дробить фреймы на мелкие области.

<frame>. Тег <frame> определяет свойства отдельного фрейма, на которые делится окно браузера.

<frameset>. Тег <frameset> заменяет собой элемент <body> на веб-странице и формирует структуру фреймов.

<iframe>. Тег <iframe> создает плавающий фрейм, который находится внутри обычного документа, он позволяет загружать в область заданных размеров любые другие независимые документы.

Значения атрибутов тегов. Атрибуты тегов расширяют возможности самих тегов и позволяют гибко управлять различными настройками отображения элементов веб-страницы. Общее количество атрибутов достаточно велико, но их значения, как правило, можно сгруппировать по разным типам, например, задающих цвет, размер, адрес и др. Далее рассмотрим основные типы значений.

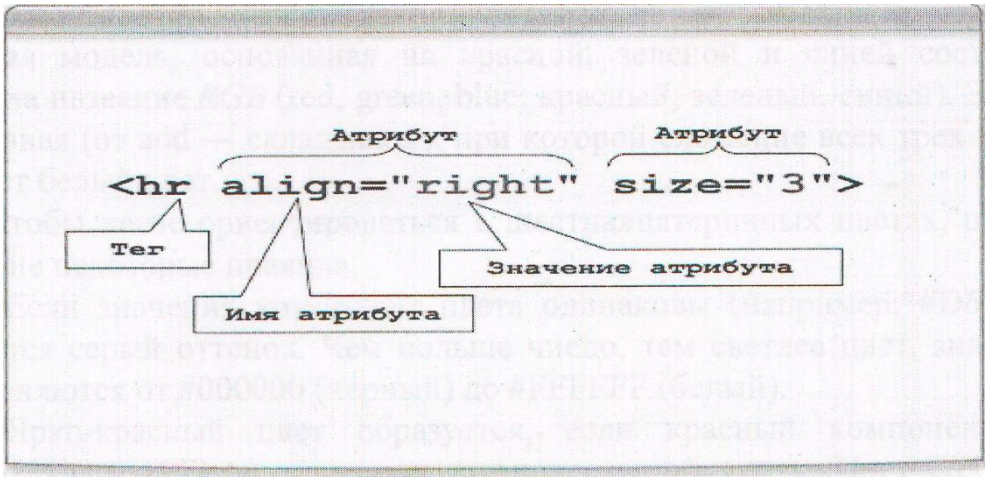

Рисунок 2.9. Значения атрибутов

Цвет. В HTML цвет задается одним из двух путей: с помощью шестнадцатеричного кода и по названию некоторых цветов. Преимущественно используется способ, основанный на шестнадцатеричной системе исчисления, как наиболее универсальный. Шестнадцатеричные цвета. Для задания цветов в HTML используются числа в шестнадцатеричном коде. Шестнадцатеричная система, в отличие от десятичной системы базируется, как следует из ее названия, на числе 16. Цифры будут следующие: 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F. Числа от 10 до 15 заменены на латинскими буквами. Числа больше 15 в шестнадцатеричной системе образуются объединением двух чисел в одно (табл.6.2). Например, числу 255 в десятичной системе соответствует число FF в шестнадцатеричной. Чтобы не возникло путаницы в определении системы счисления, перед шестнадцатеричным числом ставит символ решетки #, например, #aa69cc. При этом регистр значения не имеет, поэтому допустимо писать #F0F0F0 или #f0f0f0.

Типичный цвет, используемый в HTML, выглядит следующим образом. <bodybgcolor = “#fa8e47>. Здесь цвет фона веб-станицы задан как #FA8E47. Символ решетки # перед числом означает, что оно шестнадцатеричное. Первые две цифры (FA) определяют красную составляющую цвета, цифры с

третьей по четвертую (8E)-зеленую, а последние две цифры (47)-синюю. В итоге получится такие цвета:

Рисунок 2.10. Выбор цвет фона

Каждый из трех цветов – красный, зеленый и синий – может принимать значения от 00 до FF, что в итоге образует 256 оттенков. Таким образом, общее количество цветов может быть 256x256x256 = 16.777.216 комбинаций. Цветовая модель, основанная на красной, зеленой и синей составляющей получила название RGB (red, green, blue; красный, зеленый, синий). Эта модель аддитивная (от add - складывать), при которой сложение всех трех компонент образует белый цвет.

Чтобы легче ориентироваться в шестнадцатеричных цветах, примите во внимание некоторые правила. Если значения компонент цвета одинаковы (например,: #D6D6D6), то получится серый оттенок. Чем больше число, тем светлее цвет, значения при этом меняются от #000000 (черный) до #FFFFFF (белый). Ярко-красный цвет образуется, если красный компонент сделать максимальным (FF), а остальные компоненты обнулить. Цвет со значением #FF0000 самый красный из возможных красных оттенков. Аналогично обстоит с зеленым цветом (#00FF00) и синим (#0000FF). Желтый цвет (#FFFF00) получается смешением красного с зленым. Это хорошо видно на цветовом краге (рис.6.1), где представлены основные цвета (красный, зеленый, синий) и комплементарный или дополнительные. К ним относятся желтый, голубой и фиолетовый (еще называемым пурпурным). Вообще, любой цвет можно получить смешением близлежащих к нему цветов. Так, голубой (#00FFFF) получается за счет объединения синего и зеленого цвета.

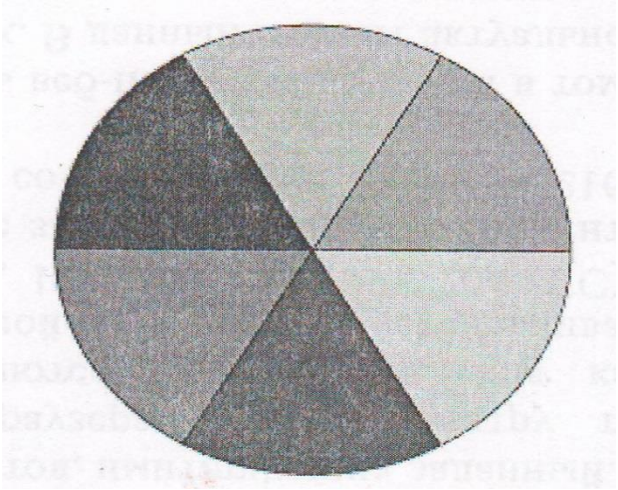

Рисунок 2.11. Цветовой круг

Цвета по шестнадцатеричным значением не обязательно подбирать эмпирическим путем. Для этой цели подойдет графический редактор, умеющий работать с разными цветовыми моделями, например, Adobe Photoshop. На рис.2.10 показано окно для выбора в этой программе, линей обведено полученное шестнадцатеричное значение текущего цвета. Его можно скопировать и вставить к себе в код.

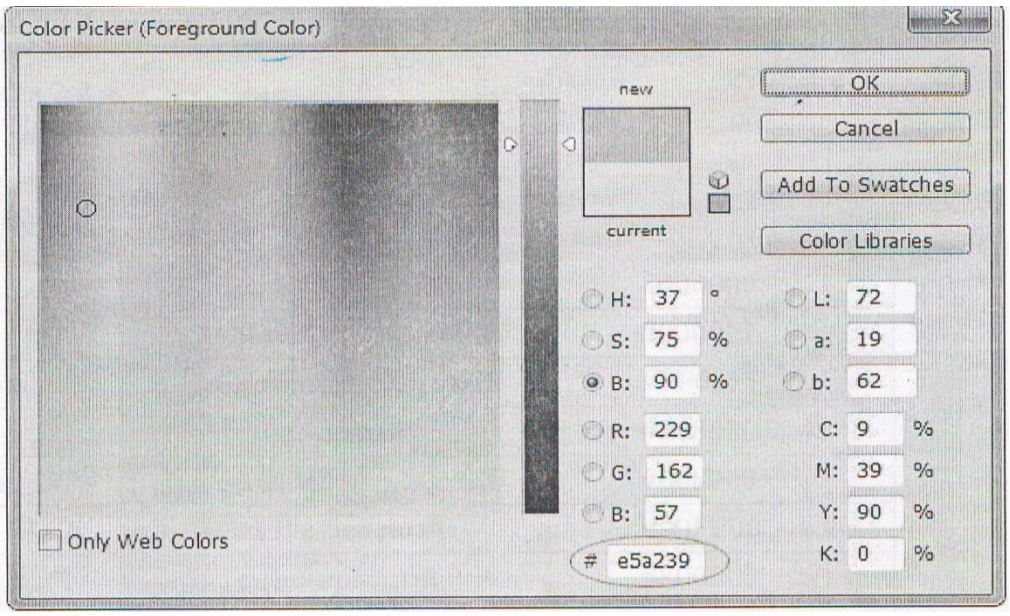

Рисунок 2.12. Окно для выбора цвета в программе Photoshop

Веб-цвета. Если установить качество цветопередачи монитора в 8 бит (256 цветов), то один и тот же цвет может показываться в разных браузерах по-своему. Это связано со способом отображения графики, когда браузер работает со своей собственной палитрой и не может показать цвет, который у него в палитре отсутствует. В этом случае цвет заменяется сочетанием пикселов других, близких к нему, имитирующих заданный. Чтобы цвет оставался неизменным в разных браузерах, ввели палитру так называемых веб-цветов. *Веб-цветами* называются цвета, для каждой составляющей которых – красной, зеленой и синей – устанавливается одно из шести значений – 0 (00), 51 (33), 102 (66), 153 (99), 204 (CC), 255 (FF). В скобках указано шестнадцатеричное значение данной компоненты. Общее количество цветов из всех возможных сочетаний дает 6x6x6 – 216 цветов. Пример веб-цвета - #33FF66.

Основная особенность веб-цвета заключается в том, что он показывается одинаково во всех браузерах. В данный момент актуальность веб-цветов весьма мала из-за повышения качества мониторов и расширения их возможностей.

Не имеет значения, каким способом вы задаете цвет – по его имени или с помощью шестнадцатеричных чисел. По своему действию эти способы равны.

Цвет фона и текста.

HTML 4.01IE 7IE 8IE 9Cr 8Op 11Sa 5Fx 3.6
<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd>
<html>
<head>
<title>Цвета</title>
<meta html-equip = “Content-Type” content = “text/html; charset = utf-8”>
</head>
<bodybgcolor = “teal” text = “#ffffff”>
Пример текста

</body>

</html>

В данном примере цвет фона задается с помощью атрибутов bicolor тега <body>, а цвет текста через атрибут text. Для разнообразия значения у атрибута text установлено в виде шестнадцатеричного числа, а у bicolor с помощью зарезервированного ключевого слова teal.

Размер. В HTML размеры элементов или расстояния между ними задаются в пикселях или процентах. *Пиксел* – это элементарная точка на экране монитора, является относительной единицей измерения, ее величина зависит от установленного экранного разрешения и размера монитора. Возьмем, к примеру, популярное разрешения монитора 1024x768 пикселов. Картинка с такими же размерами будет занимать всю область экрана. Увеличив разрешение монитора до 1280x1024, мы, тем самым, уменьшим размеры изображения на экране.

При использовании пикселов в качестве значений пишется только число без указания единиц, например, width = “380”.

Пример. Размеры изображения в пикселях

<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd >
<html>
<head>
<meta http-equip = “content-type” content = “text/html; charset=uft-8”>
<title>Изображение</title>
</head>

<body>
<imgsrc = “images/fiqure.jpg” alt=Винни-Пух в гостях у Кролика”
Width = “100” height = “111” hspace = “4” vspace = “4” border = “2”>
</body>
</html>

В данном примере рисунок имеет 100 пикселов (width = “100”), высоту 111 пикселов (height = “111”), горизонтальный и вертикальный отступ по 4 пиксела (hspace и vspace) и толщину вокруг картинки 2 пиксела (border = “2”).

Процентная запись удачно дополняет пикселы, поскольку позволяет привязаться к размеру определенного элемента, к примеру, окна браузер. Так, если задать у картинки ширину 100%, то рисунок будет заполнять все свободное пространство окна по ширине. Браузер понимает, что речь идет о процентах, если после числа добавляется символ %, например: width = “40%”.

Размеры допустимо задавать только в целых числах. Это правило относится как к пикселам, так и к процентам.

Учтите, что размер в процентах вычисляется от размеров родительского элемента, иными словами, контейнера, внутри которого располагается элемент. Если родитель явно не задан, тогда за отчет принимается окно браузера.

Пример. Размеры изображения в процентах

HTML 4.01IE 7IE 8IE 9Cr 11Op 11Sa 5Fx 4
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/strict.dtd >
<html>
<head>
<meta http-equip = “content-type” content = “text/html; charset=uft-8”>
<title>Изображение</title>
</head>
<body>
<imgsrc = “images/fiqure.jpg” alt=Винни-Пух в гостях у Кролика”
width = “100%”>
</body>
</html>

В данном примере ширина картинки задана как 100%, при этом высота изображения явно не задается, поскольку она вычисляется автоматически. Вид страницы при таких размерах картинки показан на рисунке 2.13.

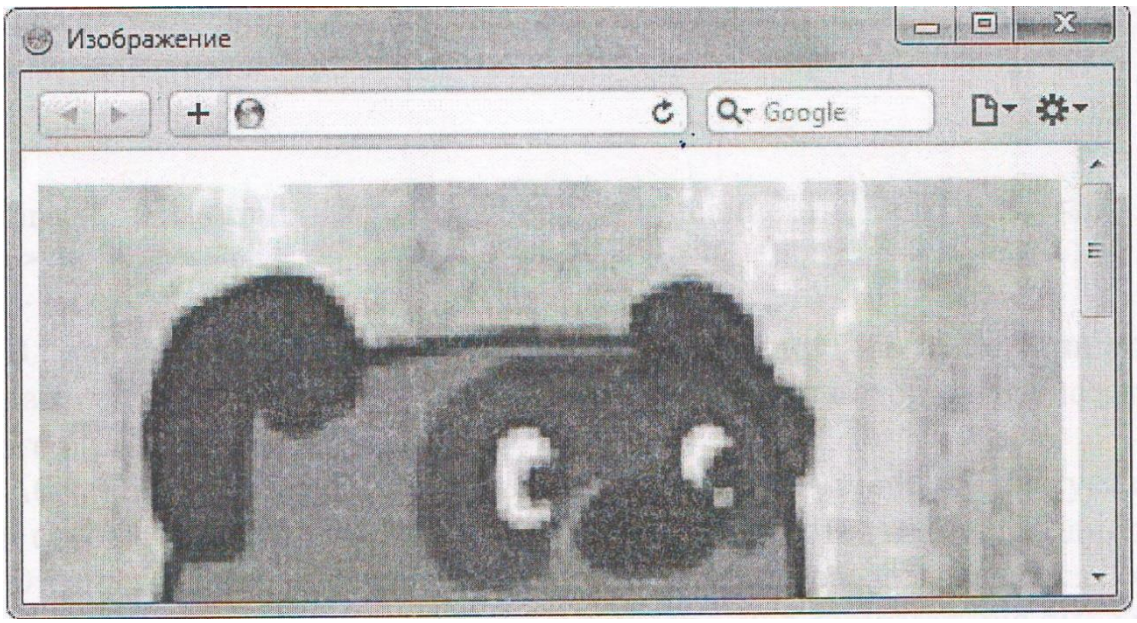

Рисунок 2.13. Изображение с шириной 100%

Обратите внимание, что в изображении появляется заметные искажения, это связано с увеличением картинки вопреки ее исходным размерам.

Как вы помните, ширина окна принимается за 100%, но ее легко превысить, причем ненароком. В частности, стоит только добавить в примере 6.3 к тегу <img> отступы по горизонтали (hspace = “10”) и ширина изображения станет 100%+20. Это в свою очередь приведет к появлению горизонтальной полосы прокрутки. Учитывайте этот нюанс при установке размеров элементов.

Адрес. Адресом называется путь к документу, например, к графическому файлу. Адрес необходим в тех случаях, когда делается ссылка на веб-страницу или загружается определенный файл. Например, в теге <img> адрес используется в качестве значения атрибута src, он задает путь к файлу с изображением.

Синонимом адреса выступает URL (Universal Resource Locator, универсальный указатель ресурсов), различают абсолютные и относительные адреса.

Абсолютные адреса. Подобные адреса работают везде и всюду независимо от имени сайта или веб-страницы, где задан URL, и начинаются всегда указания протокола передачи данных. Для веб-страниц это обычно HTTP (Hyper Text Transfer Protocol, протокол передачи гипертекста), соответственно, абсолютные адреса начинаются с ключевого слова http://. В примере 6.4 приведена ссылка, в которой применяется абсолютный адрес.

Пример. Использование абсолютного адреса в ссылке.

<! DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01//EN”
http://www.w3.org/TR/html4/strict.dtd >
<html>

<head>
<meta http-equip = “content-type” content = “text/html; charset=uft-8”>
<title>Ссылка</title>
</head>
<body>
<ahref = http://htmlbook.ru/html/body> Описание тега BODY
</body>
</html>

В данном примере текстовая ссылка ведет на сайт htmlbook.ru и указывает на веб-страницу с именем body.html, которая располагается в каталоге html.

Абсолютные адреса применяются в первую очередь для указания на другой сетевой ресурс и достаточно редко используются в рамках одного сайта.

Относительные адреса. Относительные адреса указываются от корня сайта или текущего документа. Например, код <imgsrc = “pic.gif”> означает загрузить графический файл с именем pic.gif, который располагается в той же папке, что и сама веб-страница. Далее рассмотрим несколько примеров таких адресов.

Адрес указывает обычно на файл index.html, который находится в корне сайта. Если файл index.html отсутствует, браузер, как правило, показывает список файлов, находящихся в данном каталоге. Имя файла не обязательно должно быть index.html, этот параметр меняется через настройки *веб-сервера*- так называется программа, которая анализирует приходящие от браузера запросы и передает ему документы, показываемые пользователю.

/images/pic.gif

Слеш (символ /) перед адресом говорит о том, что адресация начинается от корня сайта. Ссылка ведет на рисунок pic.gif, который находится в папке images. А та в свою очередь размещена в корне сайта.

../help/me.html

Две точки перед именем указывают браузеру перейти на уровень выше в списке каталогов сайта и там «поискать» в папке help файл me.html.

Manual/info.html

Если перед именем папки нет никаких дополнительных символов, вроде точек или слеша, то папка размещена внутри текущего каталога, а уже в ней располагается файл info.html.

Адреса относительно корня сайта вроде /demo/ работают только под управлением веб-сервера и на локальном компьютере не применимы.

Ниже приведены ссылки, в которых используются относительные адреса.

Пример. Относительные адреса в ссылках

HTML 4.01IE 7IE 8IE 9Cr 11Op 11Sa 5Fx 4
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01/EN”
http://www.w3.org/TR/html4/strict.dtd >

<html>
<head>
<meta http-equip = “content-type” content = “text/html; charset=uft-8”>
<title>Ссылки</title>
</head>
<body>
<ahref = “images/xxx.jpg”>Посмотрите на мою фотографию!
<ahref = “tip.html”>Как сделать такое же фото?
</body>
</html>

Иногда можно встретить в адресе ссылке путь в виде ./file/doc.html. Точка со слешем означает, что отсчет ведется от текущей папки. Подобная запись избыточна и ее можно сократить до file/doc.html.

Текст. Для изменения вида теста существует достаточно большое количество различных тегов. Это и немудрено, ведь текст самый популярный вид информации.

Особенности текста в HTML.

Прежде чем редактировать код веб-страницы, следует принять во внимание некоторые особенности, которые присущи HTML при работе с текстом.

Любое количество пробелов идущим подряд, в браузере отображается как один. Сколько бы вы не поставили пробелов между словами, это никак не повлияет на конечный вид текста. Это же привило относится к символам табуляции и переносу текста. Поэтому не ставьте лишних пробелов, поэтому это лишь увеличит общий объем файла, но никак не изменит вид документа в браузере. Приведенный ниже строки будут отображаться на веб-странице одинаково, несмотря на их разное написание.

Измеряй микрометром. Отмечай мелом. Отрубай топором.
Измеряй микрометром. Отмечай мелом. Отрубай топором.
Измеряй микрометром.
Отмечай мелом.
Отрубай топором.

Исключением из этого правила является тег <pre>, внутри которого любое число пробелов отображается именно так, как оно указано в коде.

Нет расстановки переносов в тексте. HTML не поддерживает расстановку переносов в словах, как это делают текстовые редакторы, иначе говоря, все слова пишутся целиком без их разбиения. Это условие несущественно, пока не используется выравнивание текста по ширине. В этом случае блок текста выравнивается по левому и правому краю. Короткие строки при этом растягиваются за счет автоматического добавления пробелов между словами. Иногда пустые блоки между словами настолько велики, что портят внешний вид страницы и ухудшают читабельность текста.

Представьте, что у вас в середине предложения есть какое-нибудь длинное слово, вот, например, «Дегидроэпиандростерон». В текстовом

редакторе это ширину, а на веб-странице подобное слово будет отображаться целиком, без переносов.

Текст занимает ширину окна браузера. Если вы просто напишите одну длинную строку в коде HTML, то в браузере будет отформатирована, чтобы текст поместился по ширине в окно. Переносы текста будут добавлены автоматически в местах пробела или дефиса. Что произойдет, если в тексте нет ни того, ни другого символа? Браузер не сможет создать переносы и отобразит текст одной строкой. Если она шире окна браузера, то неминуемо появится горизонтальная полоса прокрутки.

Абзацы. Как правило, блоки текста разделяют между собой абзацами (параграфами). По умолчанию между параграфами между параграфами существует небольшой вертикальный отступ, называемый отбивкой. Синтаксис создания абзацев следующий.

Абзац 1
Абзац 2

Каждый абзац начинается с тега и должен иметь необязательный закрывающий тег .

В любой книге для выделения следующего абзаца используется отступ первой строки, еще называемый «красная строка». Это позволяет читателю легко отыскивать взглядом новую строку и повышает, таким образом, читабельность текста. На веб-странице этот прием обычно не используется, а для разделения абзацев применяется отбивка.

В следующем примере показано применение абзацев для создания отступа между строками.

Пример. Использование абзацев.

```
 HTML 4.01IE 7IE 8IE 9Cr 11Op 11Sa 5Fx 4
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01/EN”
http://www.w3.org/TR/html4/strict.dtd >
<html>
<head>
<meta http-equip = “content-type” content = “text/html; charset=uft-8”>
<title>Применение абзацев</title>
</head>
<body>
<p>В одних садах цветет миндаль, в других метёт метель. </p>
<p>В одних краях еще февраль, в других – уже апрель. </p>
<p>Проходит время, вечный счёт: год за год, век за век…</p>
<p>Во всем на радость и печаль по двадцать пять недель. </p>
<p>Мне двадцать пять недель февраль, и двадцать пять – апрель. </p>
<p>По двадцать пять недель туман уходит счёт векам. </p>
<p>Летит мой звонкий балаган куда-то к облакам. </p>
<p><i>М.Щербаков</i></p>
</body>
</html>
```

Результат работы данного примера показан ниже.

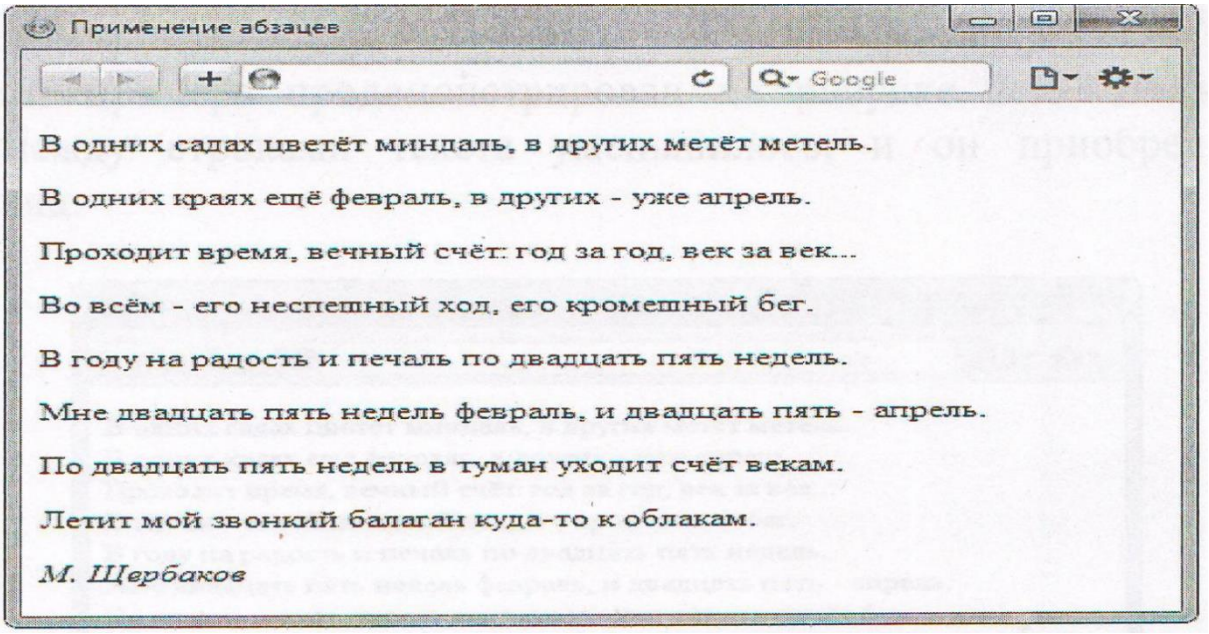

Рисунок 2.14. Отступы на веб-странице при использовании абзацев

Как видно из рисунка, при использовании тега <p> между абзацами возникают слишком большие отступы. От них можно избавиться, если в местах переноса строк добавлять тег
. В отличие от абзаца, тег переноса строки
 не создает дополнительных вертикальных отступов между строками и может применяться практически в любом тексте.

Так, текст примера с учетом переноса строк будет преобразован следующим образом:

Примет. Тег

HTML 4.0IE 7IE 8IE 9Cr 11Op 11Sa 5Fx 4

<! DOCTYPE HTML PUBLIC”-//W3C//DTD HTML 4.01//EN”
“http://www.w3.org/TR/html4/strict.dtd”>
<Html>
<Head>
<Meta http-equip = “Content-Type” content=text/html; charset=utf-8”>
<title>Переносы в тексте</title>
</head>
<body>
<p>В одних садах цветёт миндаль, в других метёт метель.
В одних краях ещё февраль, в других-уже апрель.
Проходит время, вечный счет: год за год, век за век…
Во всём-его неспешный ход, его кромешный бег.
В году на радость и печаль по двадцать пять недель.
Мне двадцать пять недель февраль, и двадцать пять-апрель.
По двадцать пять недель в туман уходит счет векам.

Летит мой звонкий балаган куда-то к облакам.
М.Щербаков
</body>
</html>

Результат примера продемонстрирован на рисунке 2.15. Видно, что расстояние между строками текста уменьшилось, и он приобрел более компактный вид.

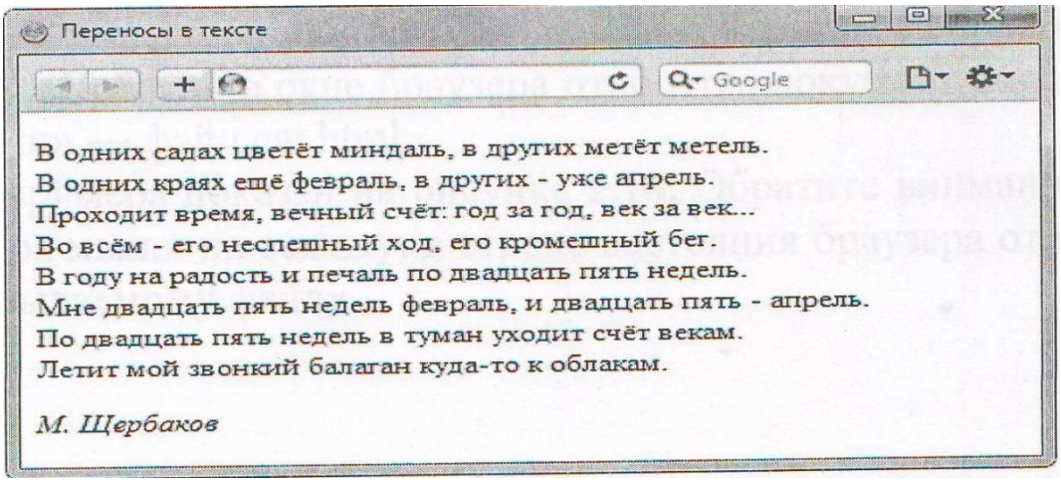

Рисунок 2.15. Вид с учетом переносов

Ссылки. Ссылки являются основой гипертекстовых документов и позволяют переходить с одной веб-страницы на другую. Особенность их состоит в том, что сама ссылка может вести не только на HTML-файлы, но и на файл любого типа, причем этот файл может размещаться совсем на другом сайте. Главное, чтобы к документу, на который делается ссылка, был доступ. Иными словами, если путь к файлу можно указать в адресной строке браузера, и файл при этом будет открыт, то га его можно сделать ссылку.

Для создан ссылки необходимо сообщить браузеру, что является ссылкой, а также указать адрес документа, на который следует сделать ссылку. Оба действия выполняются с помощью тега <a>. Общий синтаксис создания ссылок следующий.

<ahref = “URL”> текст ссылки

Атрибут href определяет URL (Universal Resource Locator, универсальный указатель ресурса), иными словами, адрес документа, на который следует перейти, а содержимое контейнера <a> является ссылкой. Текст, расположенный между тегами <a> и , по умолчанию становиться синего цвета и подчеркивается. В данном примере показано создание нескольких ссылок на разные веб-страницы.

Пример. Добавление ссылок
HTML 4.01IE 7IE 8IE 9Cr 11Op 11Sa 5Fx 4
<DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01//EN”

“http://www.w3.org/TR/html4/strict.dtd”>
<Html>
<Head>
<Meta http-equip = “content-type” content = “text/html; charset=uft-8”>
<title>Ссылки на странице</title>
</head>
<Body>
<ahref = “dog.html”>Собаки
<ahref = “cat.html”>Кошки
</body>
</html>

В данном примере создаются две ссылки с разными тестами. При щелчке по тексту «Собаки» в окне браузера откроется документ dog.html, а при щелчке на «Кошки» - файл cat.html.

Результат примера показан на рисунке 2.14. Обратите внимание, что при наведении курсора мыши на ссылку, в строке состояния браузера отображается полный путь к ссылаемому файлу.

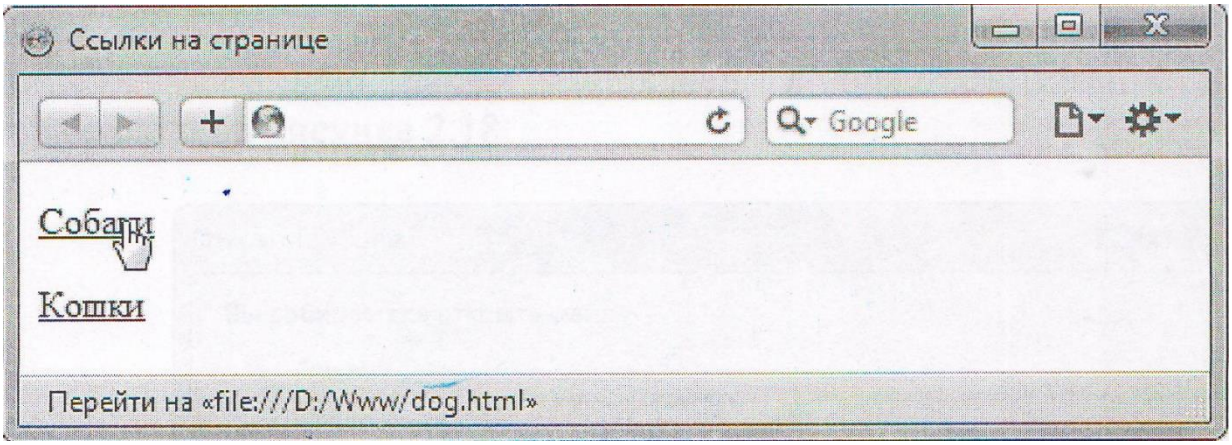

Рисунок 2.16. Вид ссылок на странице

Если указана ссылка на файл, которого не существует, например, его имя в атрибуте href набрано с ошибкой, то такая ссылка называется *битая*. Битых ссылок следует категорически избегать, поскольку они водят посетителей сайта в заблуждение. Так, при щелчке по ссылке из вышеуказанного примера в браузере Safari откроется не сам документ, а окно с предупреждением на рисунке 2.16.

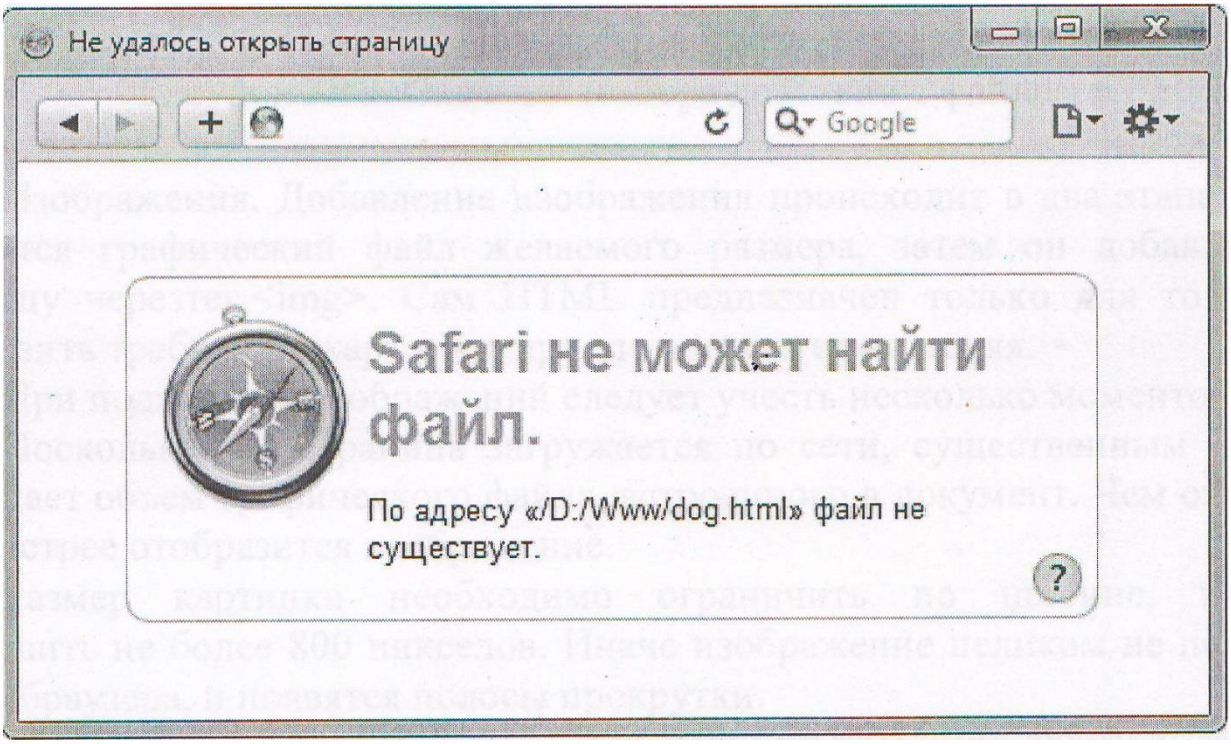

Рисунок 2.17. Результат при открытии битой ссылки

Естественно, подобное сообщение будет различаться в браузерах, но смысл останется один – документ, на который ведет ссылка, не может быть открыт. Чтобы не возникло подобных ошибок, тестируйте все ссылки на их работоспособность и сразу же устраняйте имеющиеся погрешности.

Файл по ссылке открывается в окне браузера только в тех случаях, когда браузер знает тип документа. Но поскольку ссылку можно сделать на файл любого типа, то браузер не всегда может отобразить документ. При этом выводиться сообщение, как следует обработать файл – открыть его или сохранить в указанную папку. Например, в браузере Firefox выводится следующее окно на рисунке 2.18.

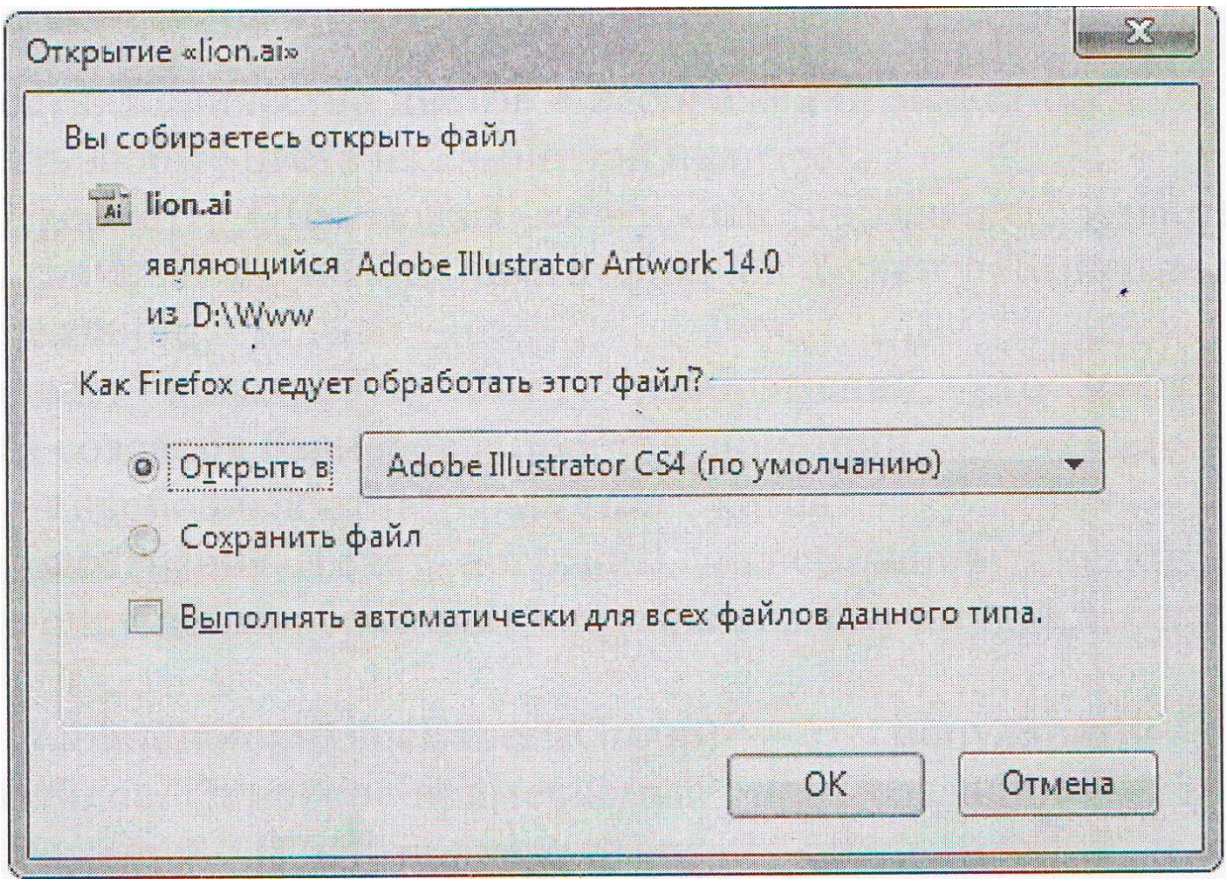

Рисунок 2.18. Окно для выбора действия с файлом в Firefox

Изображения. Добавление изображения происходит в два этапа: вначале готовится графический файл желаемого размера, затем он добавляется на страницу через тег <img>. Сам HTML предназначен только для того, чтобы отобразить требуемую картинку, при этом никак ее не меняя.

При подготовке изображений следует учесть несколько моментов.

Поскольку веб-страница загружается по сети, существенным фактором выступает объем графического файла, встроенного в документ. Чем он меньше, тем быстрее отобразится изображение.

Размер картинки необходимо ограничить по ширине, например, установить не более 800 пикселов. Иначе изображение целиком не поместится в окне браузера, и появятся полосы прокрутки.

Широкое распространение для веб-графики получили два формата – GIF и JPEG. Их многофункциональность, универсальность, небольшой объем исходных файлов при достаточном для сайта качестве, сослужили им положительную службу, фактически определив их как стандарт веб-изображений. Есть еще формат PNG, который также поддерживается браузерами при добавлении изображений и существует в двух ипостасях – PNG-8 и PNG-24. Однако популярность PNG сильно уступает признанию форматов GIF и JPEG.

Формат GIF.

GIF (Graphics Interchange Format) – формат графических файлов, широко применяемый при создании сайтов. GIF использует 8-битовый цвет и эффективно снижает сплошные цветные области, при этом сохраняя детали изображения.

Особенности. Количество цветов в изображении может быть от 2 до 256, но это могут быть любые цвета из 24-битной палитры.

Файл в формате GIF может содержать прозрачные участки. Если используется отличный от белого цвета фон, он будет проглядывать сквозь «дыры» в изображении.

Поддерживает покадровую смену изображений, что делает формат популярным для создания баннеров и простой анимации.

Использует свободный от потерь метод сжатия.

Область применения. Текст, логотипы, иллюстрации с четкими краями, анимированные рисунки, изображения с прозрачными участками, баннеры.

Формат JPEG.

JPEG (Joint Photographic Expert Group) – популярный формат графических файлов, широко применяемый при создании сайтов и для хранения изображений. JPEG поддерживает 24-битовый цвет и сохраняет яркость и оттенки цветов в фотографиях неизменными. Данный формат называют сжатием с потерями, поскольку алгоритм JPEG выборочно отвергает данные. Метод сжатия может внести искажения в рисунок, особенно содержащий текст, мелкие детали или четкие края. Формат JPEG не поддерживает прозрачность. Когда вы сохраняете фотографию в этом формате, прозрачные пиксели заполняются определенным цветом.

Особенности. Количество цветов в изображении – около 16 миллионов, что вполне достаточно для сохранения фотографического качества изображения. Основная характеристика формата – качество, позволяющее управлять конечным размером файла. Поддерживает технологию, так называемый прогрессивный JPEG, в котором версия рисунка с низким разрешением появляется в окне просмотра до полной загрузки самого изображения.

Область применения. Используется преимущественно для фотографий. Не очень подходит для рисунков, содержащих участки, мелкие детали или текст.

Формат PNG-8/

PNG (Portable Network Graphics) – формат по своему действию аналогичен GIF. По заверению разработчиков использует улучшенный формат сжатия данных, но как показывает практика, это не всегда так.

Особенности. Использует 8-битную палитру (256 цветов) в изображении, за что и получил в своем названии цифру восемь. При этом можно выбирать, сколько цветов будет сохраняться в файле – от 2 до 256. В отличие от GIF, не отображает анимацию ни в каком виде.

Область применения. Текст, логотипы, иллюстрации с четкими краями.

Формат PNG-24.

PNG-24 - формат, аналогичный PNG-8, но использующий 24-битную палитру цвета подобно формату JPEG, сохраняет яркость и оттенки цветов в фотографиях. Подобно GIF и формату PNG-8, сохраняет детали изображения, как, например, в линейных рисунках, логотипах, или иллюстрациях.

Особенности. Использует примерно 16,7 млн. цветов в файле, из-за чего этот формат применяется для полноцветных изображений.

Поддерживает многоуровневую прозрачность, это позволяет создавать плавный переход от прозрачной области изображения к цветной, так называемый градиент.

Из-за того, используемый алгоритм сжатия сохраняет все цвета и пикселы в изображении неизменными, если сравнивать с другими форматами, то у PNG-24 конечный объем графического файла получается наибольшим.

Область применения. Фотографии, рисунки, содержащие прозрачные и полупрозрачные участки, рисунки с большим количеством цветов и четкими краями изображений.

Браузеры.

Разработчики браузеров не всегда следуют спецификации и в некоторых случаях трактует код не по заданным правилам, а по-своему. В конечном итоге это приводит к тому, что веб-страница, которая правильно (т.е. так, как и задумывали разработчики) отображается в одном браузере, выводится с ошибками в другом. Следовательно, спецификации в подобных случаях, скорее всего, отпугнет пользователей некоторых браузеров. К примеру, Internet Explore (IE) в настоящее время занимает лидирующее положение среди браузеров, но при этом поддерживает спецификацию HTML и CSS хуже, чем Firefox иOpera. Очевидно, что пользователи IE при посещении сайта выполненного по всем стандартам, но не учитывающего специфику этого браузера, увидят неприглядную картину.

Заказчикам сайта, а также их разработчикам подобная ситуация не по нраву, по этому стоя перед выбором: стандарты или браузер, они в большинстве своем выбирают браузер.

Получается неутешительная картина – тратить время на отладку кода для соответствия спецификации нет особой нужды. Это время лучше посвятить тому, чтобы документ без проблем работа в разных браузерах – так в основном размышляют веб-разработчики.

Резюме.

Так стоит ли проводить валидацию документов и заниматься этим этапом при написании веб-страницы? Безусловно, да. Кому- то она поможет выявить существующие недочеты, другому поможет писать корректный код. Исправлять же ошибки, добиваясь полного соответствия стандартам, или оставить их ради совместимости с разными браузерами – здесь уже каждый решает сам, какие цели он преследует и что для него важнее.

Для проверки веб-страниц на наличие ошибок и замечаний существует множество путей и способов. Условно они делятся на онлайновые и локальные. Онлайновые предназначены для проверки страниц с помощью браузера через Интернет, а локальные используются для проверки документов на текущем компьютере. Далее рассмотрим популярные методы валидации документов.

validator.w3.org

По адресу http://validator.w3.org располагается, пожалуй, самый распространённый инструмент для проверки отдельных страниц на валидность. Этот сайт предлагает три способа проверки: по адресу, локального файла и введенного в форму кода.

Проверка по адресу. Если ваш сайт уже опубликован в Интернете, то любую страницу можно проверить, вводя в текстовое поле ее адрес как указано на рисунке 2.19.

Так, вводя http://htmlbook.ru в форме «Validate by URL» (валидация по адресу) и нажав кнопку Check (проверить) получим сообщение о том, валидный документ или нет.

Хотя в текстовом поле водится адрес сайта, проверяется не сайт целиком, а только одна главная страница. Учтите, что, к примеру, адрес http://htmlbook.ru равнозначен вводу http://htmlbook.ru/index.php.

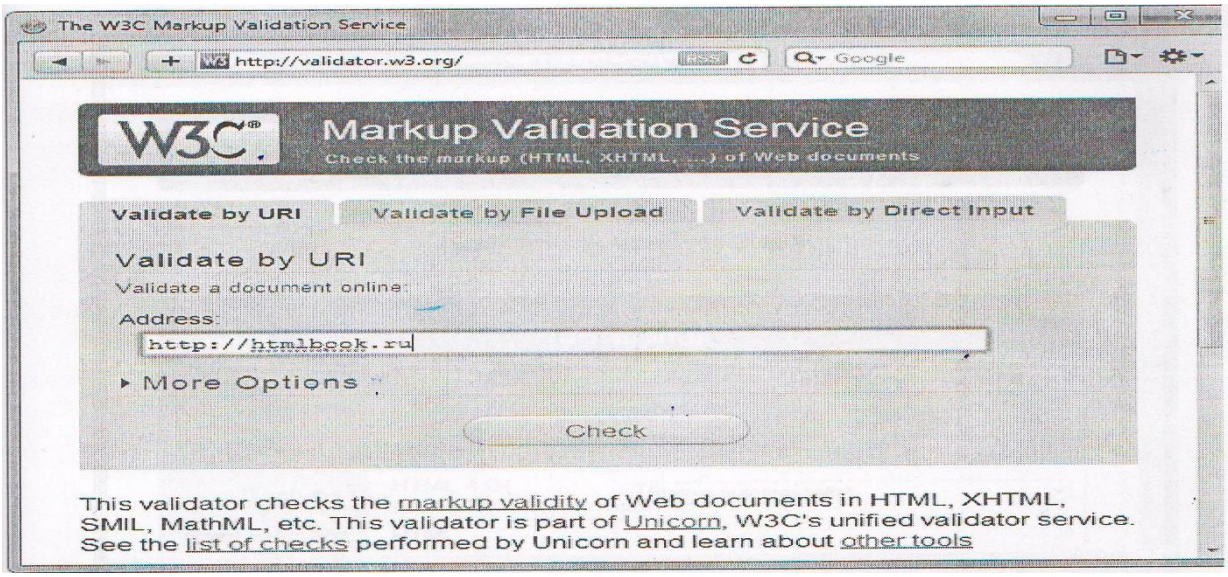

Рисунок 2.19. Форма для ввода адреса документа

Валидатор проверяет HTML-код страницы и в случае отсутствия ошибок докладывает о валидности документа как указано на рисунке 2.20

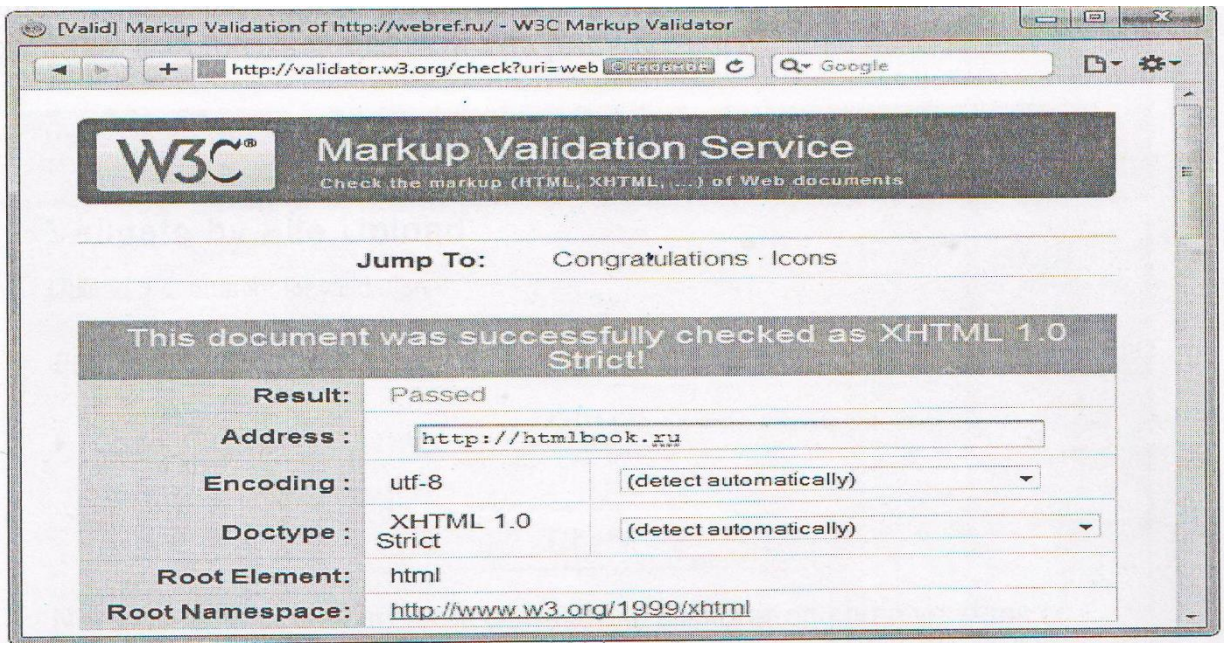

Рисунок 2.20. Отчет о проверке и валидности веб-страницы

При обнаружении ошибок выводится уведомление о том, что страница не валидна и список ошибок с указанием строк, где встречаются ошибки указано на рисунке 2.21.

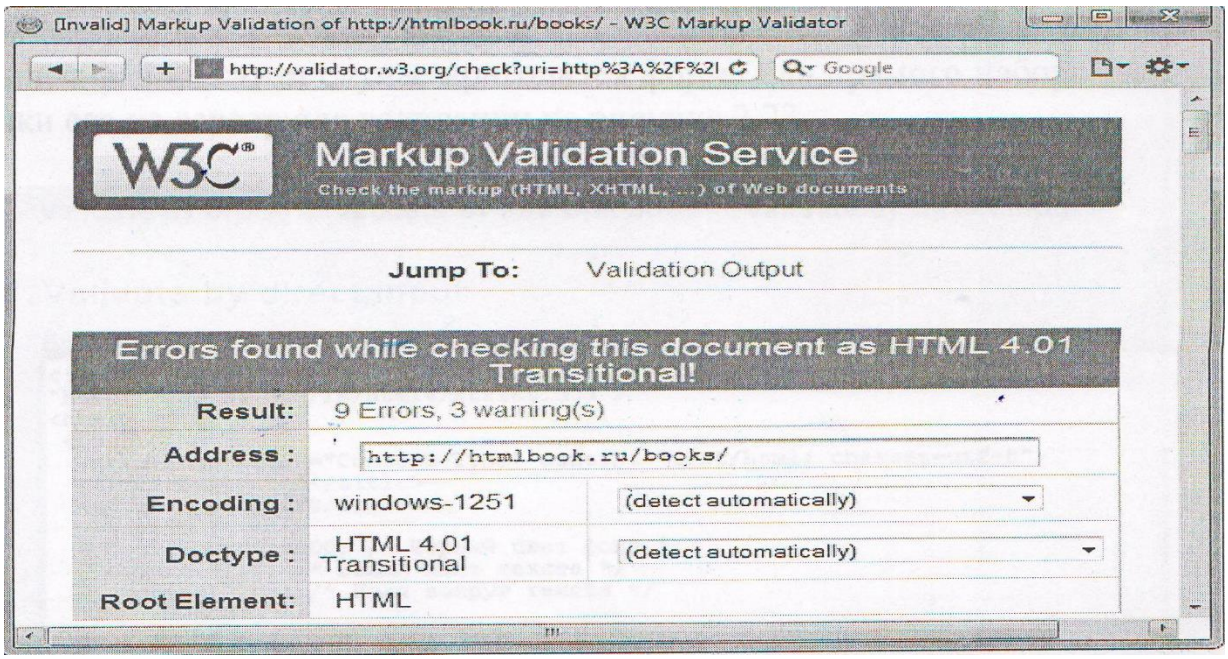

Рисунок 2.21. Отчет о проверке и вывод ошибок

Проверка локальных файлов.

Документы, еще не выставленные в Интернете, можно проверить с помощью формы, озаглавленной «Validateby File Upload» (валидация загруженных файлов), как показано на рисунке 2.22.

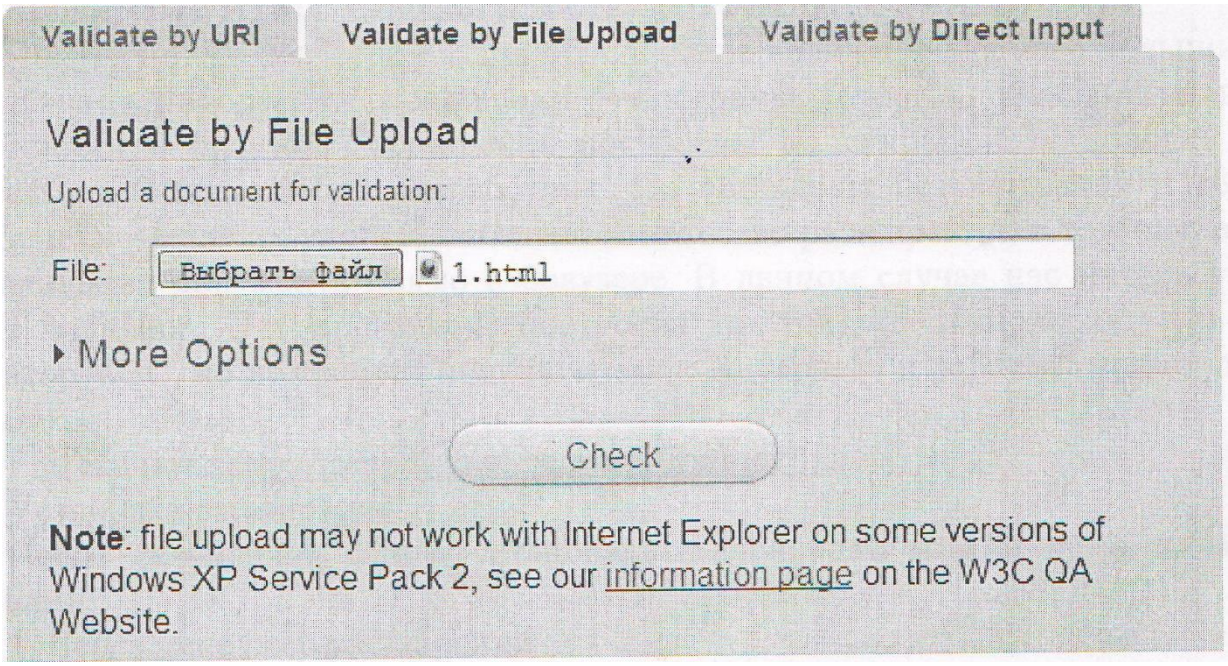

Рисунок 2.22. Форма ввода пути к локальному файлу для его проверки

Вначале следует указать путь к HTML-файлу, после чего нажать кнопку Check. Файл будет загружен на сервер и проверен на ошибки.

Использование формы для ввода кода.

В некоторых случаях требуется проверить код без сохранения его в отдельный файл. В этом случае пригодится форма для прямого набора текста и отправки его на сервер для валидации в рисунке 2.23.

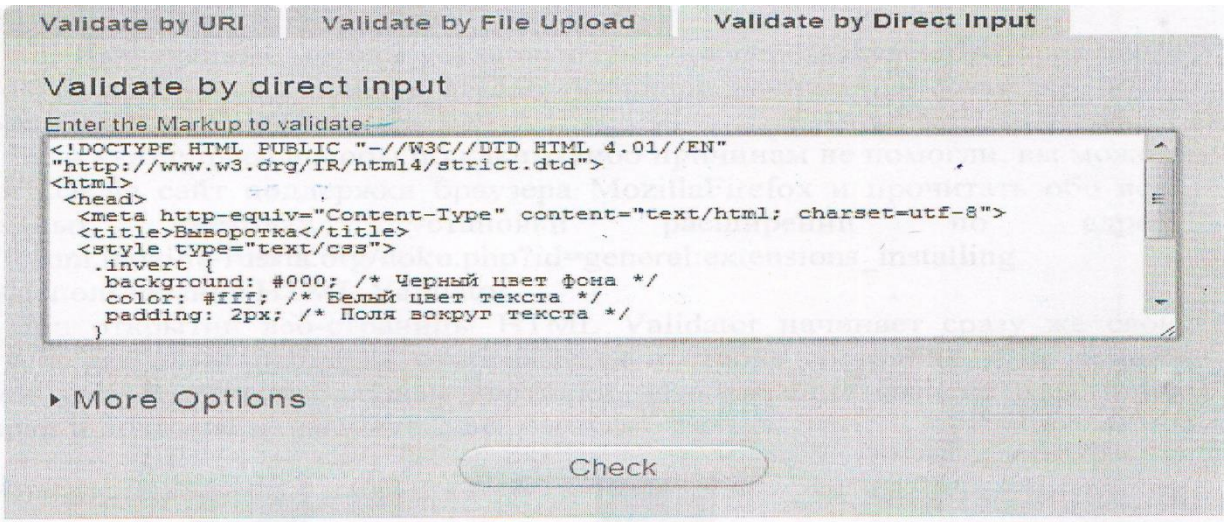

Рисунок 2.23. Форма для ввода HTML-кода

Расширения HTML Validator для браузера Firefox.

Популярность Firefox обусловлена наличием для него большого количества разнообразных – программ, которые добавляют новые возможности в браузер. Расширения построены по открытой технологии и написать их может любой разработчик. Не оставлены без внимания и веб-разработчики – для их удобства создано множество расширений, в том числе и для валидации документа прямо в браузере. В данном случае нас интересует HTML Validator. Эта программа построена по той же технологии, что и валидатор W3C, но не требует подключения к Интернету и работает прямо «на лету».

Использование HTML Validator.

При открытии веб-страницы HTML Validator начинает сразу же свою работу, и результат проверки отображается в строке состояния, в ее правом нижнем углу в виде небольшой картинки. Изображение зависит от статуса проверки и показано на рисунке 2.24.

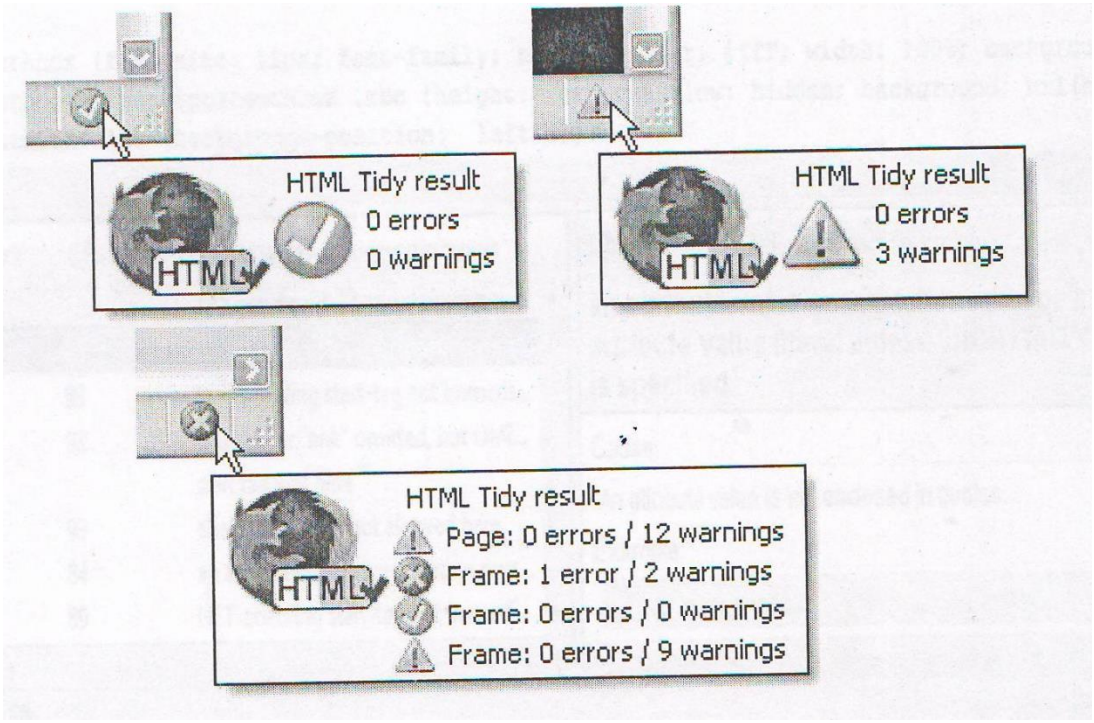

Рисунок 2.24. Виды картинок, отображаемых при проверке документа

Зеленый кружок с галочкой на рисунке 2.24 показывает, что документ валидный, желтый треугольник с восклицательным знаком на рисунке 2.24 – по коду имеются замечания, которые могут быть исправлены автоматически. А красный кружок с крестиком на рисунке 2.24 предупреждает, что есть серьезные ошибки.

Просмотреть все ошибки можно двумя способами. Во-первых, заглянуть в HTML-код документа через меню Вид> Исходный код страницы

или щелкнуть правой кнопкой и в контекстном меню выбрать Просмотр исходного кода страницы.

Окно исходного кода веб-страницы разделено на три части, верхний блок содержит собственно HTML-код. В левом нижнем блоке отображается список ошибок и замечаний или информационные сообщения в случае валидного документа. Правый нижний блок предназначен для подробных подсказок о текущих замечаниях.

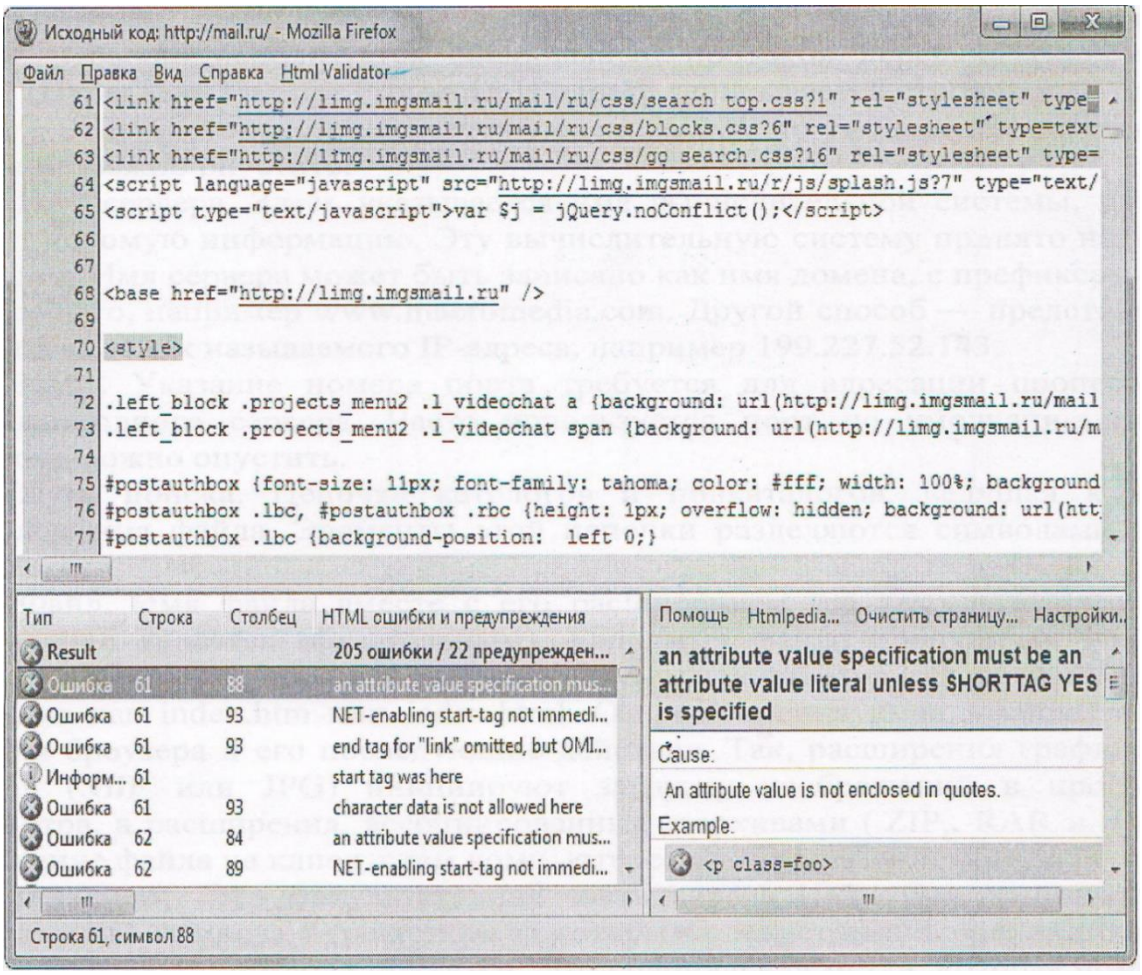

Рисунок 2.25. Результат работы расширения HTML Validator

2.4 Адресация для WWW

Для адресации ресурсов во всемирной паутине используются так называемый адреса URL. Адрес URL записывается по старому формату. Основные составляющие адрес. Протокол. Часть, которая описывает способ связи клиентов и серверов WWW. Обычно используется специальный протокол передачи гипертекста HTML. В адресе записывается строчными буквами (http://). Этот протокол принимается в адресе WWW по умолчанию, поэтому его описание можно опустить.

Имя сервера. Здесь указывается имя вычислительной системы, которая хранит искомую информацию. Эту вычислительную систему принято называть сервером. Имя сервера может быть описано как имя домена, с префиксом www.macromedia.com. Другой способ – представление имени в виде так называемого IP-адреса 199.227.52.143.

Порт поиска. Цепочка каталога и подкаталогов, ведущая к месту расположения файла. Элементы этой цепочки разделяются символами косой черты /.

Файл. Имя файла вместе с его расширением (на тех платформах, где расширение является обязательным). Если имя файла пропущено, то Web-браузер ищет файл, предоставленный по умолчанию. Имя такого файла часто выглядит как index.htm или index.html. От расширения имени файла зависит реакция браузера и его последующие действия. Так, расширения графических файлов (.GIF или JPG) инициируют загрузку изображений в программу просмотра, а расширения, ассоциированные с архивами (. ZIP, RAR и пр.), - сохранение файла на клиентском компьютере.

Фрагмент. Это факультативная часть адреса URL. Она начинается со специального символа # (решетка), за которым следует имя якоря, или, как его иногда называют, символами привязки. Это специальная метка, которая помечает фрагмент гипертекстового документа. Введение в состав имени символа привязки позволяет не ко всей странице, а к определенной ее части.

Связь по протоколу HTTP является не единственной, хотя и самой распространенной возможностью в сети Интернет. Адреса объектов Интернета могут включать в себя и другие способы связи и протоколы.

Одной из причин, объясняющих сегодняшнюю популярность WWW, является потенциал протокола HTTP. Некоторые протоколы предыдущих поколений он поглощает целиком, а некоторые просто вытесняет, предоставляя лучшие возможности, поддержанные наглядным графическим интерфейсом.

3 Описание программной части

Работа над сайтом для научно-исследовательского центра началась с инсталляционного файла install.php, через него и настраивается файл config.php. Файл install.php нужен для сохранения данных о сайте и базе данных. После инсталляции в файл config.php заносятся общие переменные, данные администратора, осуществляется соединение с сервером баз данных. При запуске Apache, в строке вписывается http://local host/.../ (вместо точек название папки с сайтом), после чего в окне появляется меню установки сайта. Именно с помощью install.php открывается это окно установки сайта.

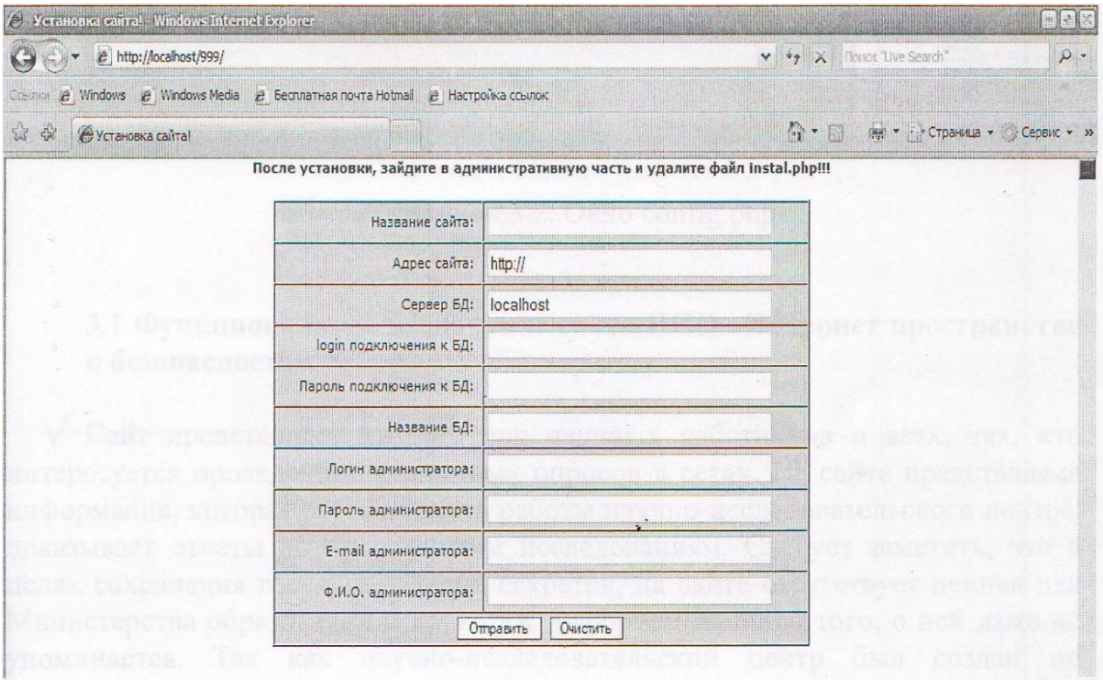

Рисунок 3.1. Окно установки сайта

После установки следует зайти в административную часть и удалить файл install.php. А заполненные мною данные сохранились в config.php. «Подсветка» кода помогает не запутаться в среде десяток строк, к тому же Dreamweaver MX распознает программные блоки, и их можно сворачивать, нажав на кнопочку «-» (между номером строки и самой строкой).

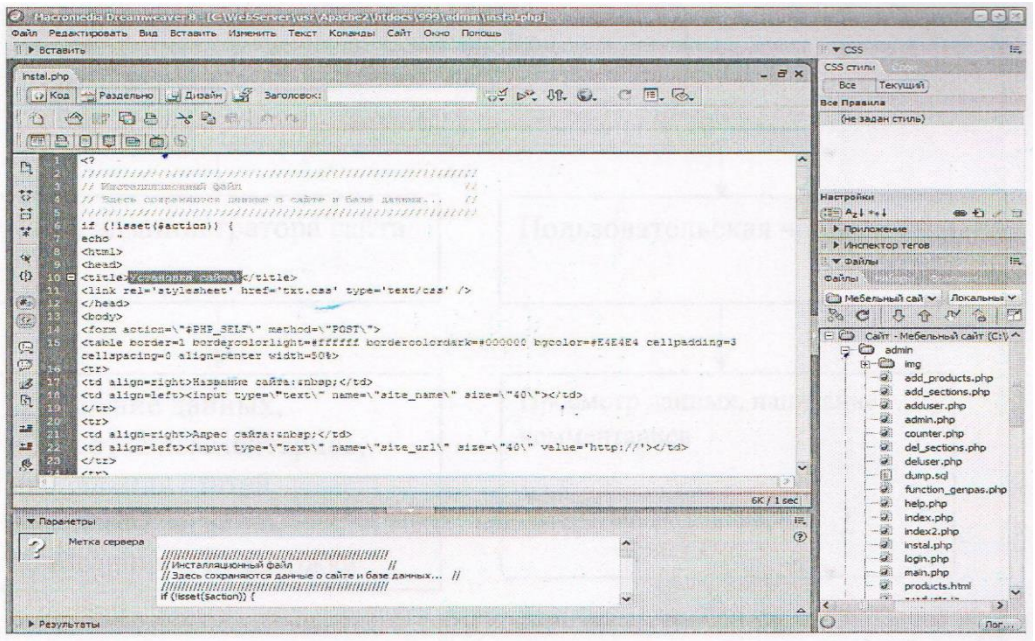

Рисунок 3.2. Окно config.php

3.1 Функциональное назначение сайта НИЦ «Интернет пространство и безопасность»

Сайт представляет интерес для научных работников и всех, тех, кто интересуется проведением различных опросов в сетях. На сайте представлена информация, которая отражает суть работы научно-исследовательского центра, показывает отчеты по проведенным исследования. Следует заметить, что в целях сохранения государственных секретов, на сайте отсутствует ценная для Министерства образования и науки РК информация. Мало того, о ней даже не упоминается. Так как научно-исследовательский центр был создан по протекции МОН РК, отчеты, представленные на сайте выложены с разрешения министерства.

3.2 Логическая структура сайта

Для удобства пользователей, сайт не подвержен перезагруженности графическим материалом. Для того, чтобы сайт и в дальнейшем вызывал интерес у пользователе, сайт логически поделен на две части – пользовательская и административная.

3.3 Функциональные требования к системе

При работе с сайтом необходимо следующие технические средства для более эффективной производительности.

Основные требования и характеристики:

1) Процессор (CPU): Intel Core 2 Duo 3.0 G
2) Оперативная память (ОЗУ): 2048 mb
3) Носитель информации (HDD): 360 Gb
4) Сетевой адаптер (LAN): 100/100 mb
5) Операционная система: Windows 2010 Server
6) СУБД: MS SQL Server 200

3.4 Входные и выходные данные

Для запуска сайта следует запустить Денвер, нажав ярлык Start Denser на рабочем столе. После чего набрать в адресной строке название сайта – nick.kz. На рисунке 3.3 показано главное окно сайта.

Главная страница выглядит следующим образом:

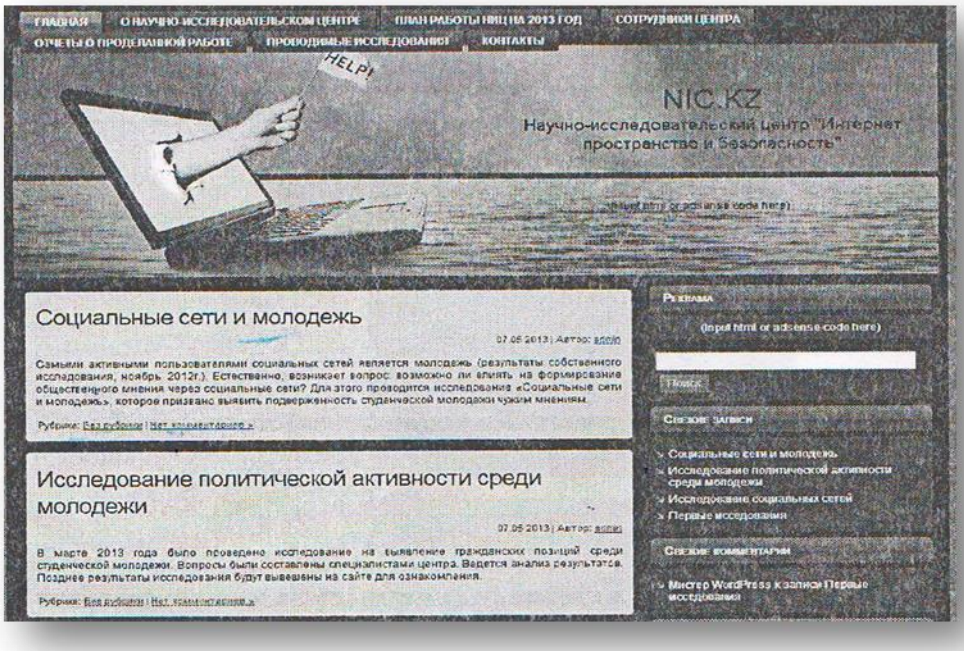

Рисунок 3.4. Главное окно сайта

Внешний вид сайта отличается выбранной цветовой гаммой соответствующей теме. Для его создания был сконструирован простейший вид.

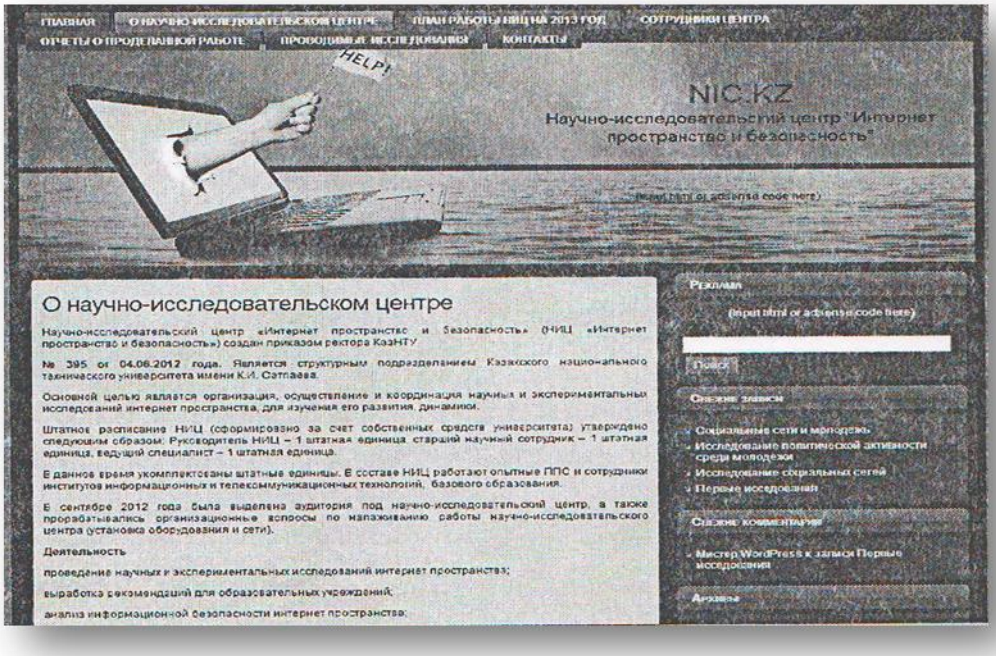

Рисунок 3.5. Страница «О научно-исследовательском центре»

С правой стороны и в верхней части на каждой странице данного сайта находится панель навигации или карта сайта, облегчающая переход по страницам. Панель состоит из ссылок, позволяющих перейти на нужную страницу пользователю. В меню находится ссылка на страницы.

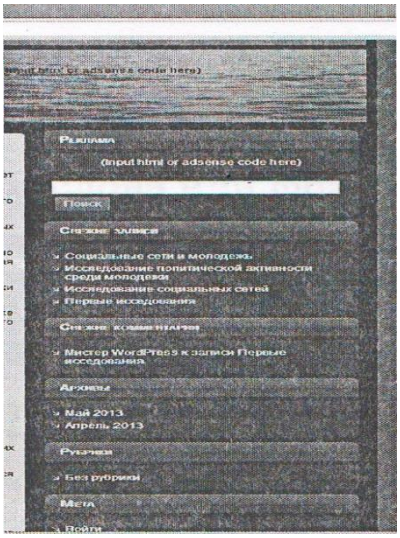

Рисунок 3.6. Боковая панель навигации по сайту

Главный принцип работы панели навигации – гиперссылка. Гиперссылки позволяют перейти к другому разделу текущего документа или Web-страницы, к другому документу Word или другой Web-странице или к файлу, созданному в другой программе. С помощью гиперссылок можно переходить также к файлам мультимедиа, в том числе звукозаписям и видеозаписям, скачиванием информации и т.д. Навигация — это механизм, который позволяет посетителю найти то, что вы хотите ему показать.

Код меню навигации по сайту:

HTML код реализации раздела навигации по сайту:

<td valign = “top” width = “150”><table width = “150” border = “1” cell padding = “0” cell spacing = “0” border color = “#FFFFFF”>

<TR>

<td width = “150” class = “zag_txt” valign = “top” border color = “#000000” bgcolor = “#A4EB9B” background = “img/zag.gif”>Навигация</td>

</TR>

<TR>

<td style = “padding-left: 5px;” class = “txt” border color = “#000000” bgcolor = “#F9FFF5”>

<a href = “#” ID = “Star Menu_0” on Mouse Over = “show (“menua”,0); clear Timeout (timeout)” on Mouse Out = “timeout =set Timeout (“hideA11()”,500);”>

<td>

</tr>

</table>

</td>

На сайте предусмотрен поиск информации. Для этого в боковой панели имеется поле для ввода требуемого слова или фразы.

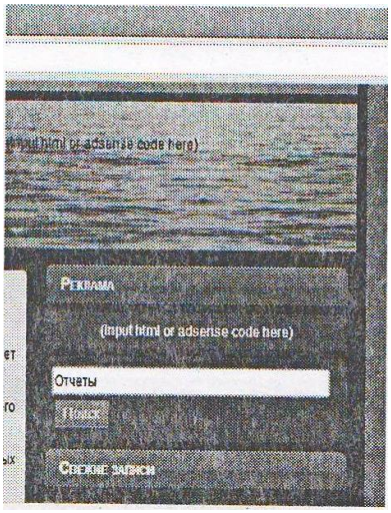

Рисунок 3.7. Поиск информации по сайту

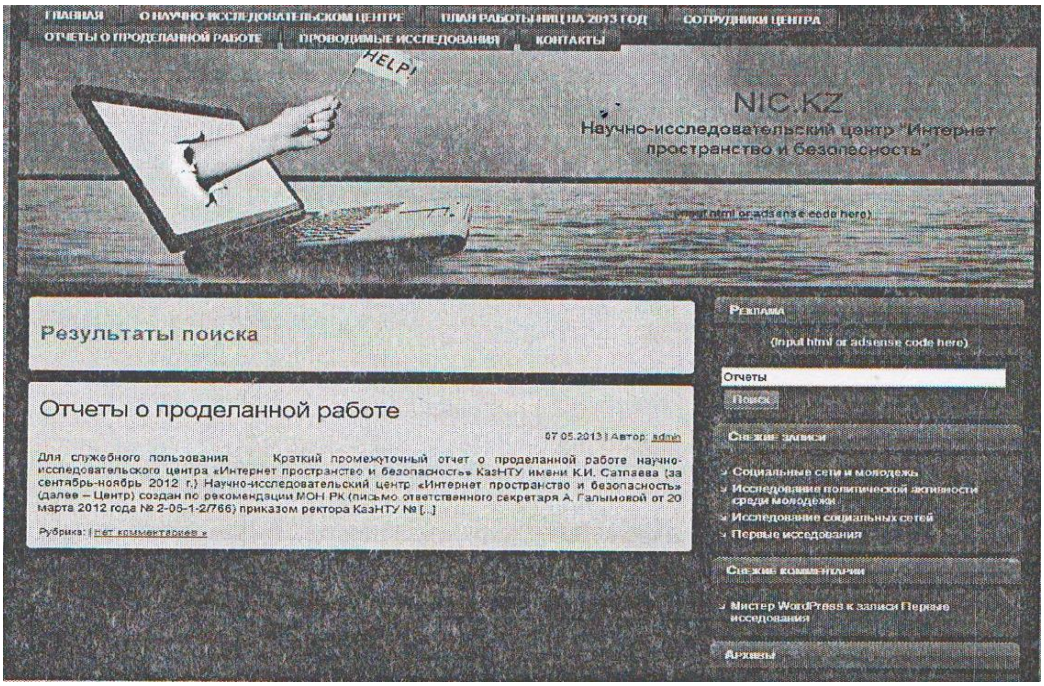

Рисунок 3.8. Результат поиска

4. Техники–экономическое обоснование

4.1 Краткая характеристика проекта

Цель дипломного проекта Разработка сайта научно-исследовательского центра «Интернет пространство и безопасность». С внедрением прикладного приложения повысится количество клиентов и спрос на предоставляемые работы. Наиболее важным моментом для разработчика, с экономической точки зрения, является процесс формирования цены. На разработку программного продукта средней сложности обычно требуются весьма незначительные средства.

В данном разделе проводится расчет цены ПП в организационно-экономической части дипломной работы предлагается производить следующим образом:

- если ПП разработан одной организаций по заказу другой и не предназначен для тиражирования, то затраты на разработку ПП считаются его себестоимость, и при формировании цены применяется затратный метод;
- если ПП предназначен для тиражирования, то конечная цена определяется путем экспертных оценок на основании ценностного подхода с учетом текущих цен конкурентов (если аналогичные ПП).

4.2 Расчет затрат на разработку информационных технологий

Под информационными технологиями понимаются экономические информационные системы (ЭИС), программные продукты (ПП), информационные базы данных и т.д.

Расчет полных затрат на разработку проектного решения в виде информационных технологий ($С_{пі}$) осуществляется по формуле

$$С_{пі} = З_{фот} + З_{юзя} + М_і + А + Р_{мі} + П_{зі} + Р_{ні}, \qquad (4.1)$$

где $З_{фот}$– общий фонд оплаты труда разработчиков, тенге;

$З_{сзі}$– отчисления по социальному налогу, тенге;

$М_і$– затраты на материалы, тенге;

А–амортизация;

$Р_{мі}$– затраты, связанные с эксплуатацией техники, тенге;

$П_{зі}$– прочие затраты, тенге;

$Р_{ні}$– накладные расходы, тенге.

Размер фонда оплаты труда разработчиков ($З_{ФОТ}$) рассчитывается по формуле

$$З_{ФОТ} = З_{оі} + З_{ді}, \qquad (4.2)$$

где $З_{оі}$ – основная заработная плата, тенге;

$З_{ді}$ – дополнительная заработная плата, тенге.

Затраты на оплату труда зависят от объема и трудоемкости разработки программного обеспечения.

Общий объем (V_0) программного продукта определяется исходя из количества и объема функции, реализуемых программой

$$V_0 = \sum_{j=1}^{n} V_i \ , \qquad (4.3)$$

где – объем отдельной функции ПО;

n – общее число функций.

Расчет уточненного объема ПО представлен в таблице 4.1.

$V_0 \approx 9525$

Таблица 4.1 - Перечень и объем функций программного модуля

№ функции	Наименование (содержание)	Объем функции (LOC)	
		по каталогу V_i	уточненный V_{yi}
1	2	3	4
201	Генерация структуры базы данных	4400	1050
203	Формирование баз данных	2180	900
204	Обработка наборов и записей баз данных	2690	565
206	Обслуживание баз, данных в интерактивном режиме	7000	550
207	Манипулирование данными	9550	3900
208	Организация поиска и поиск в базе данных	5780	1260
707	Графический вывод результатов	480	1300
	Итого:	32080	9525

Общая трудоемкость небольших проектов рассчитывается по формуле

$$T_0 = T_н \cdot К_с \cdot К_т \cdot К_н \ , \qquad (4.4)$$

где $T_н$ – нормативная трудоемкость;

$К_с$ – коэффициент, учитывающий сложность ПО;

$К_т$ – поправочный коэффициент, учитывающий степень использования при разработке стандартных модулей;

$К_н$ – коэффициент, учитывающий степень новизны ПО.

Посредством коэффициента сложности, показанный в таблице 4.2, учитывается затраты труда, связанные со сложностью ПП [4].

В разрабатываемом проекте $К_с$, за счет наличия у программного модуля одновременно двух характеристик:

- режим работы в реальном времени;
- управление удаленными объектами;

Принимаем $К_с$= 0,19 из таблицы 4.2.

Коэффициент, учитывающий степень использования при разработке ПО стандартных модулей ($К_т$). Степень использования в разрабатываемом ПО стандартных модулей определяется их удельным весом в общем объеме проектируемого продукта. В данном проекте степень охвата реализуемых функций разрабатываемого ПО стандартными модулями, типовыми программами и ПО от 20 % до 40 %, следовательно, = 0,8.

Таблица 4.2 - Дополнительные коэффициенты сложности ПО

Характеристика ПО	Значения $К_с$
1. Функционирование ПО в расширенной операционной среде (связь с другими ПО)	0,05
2. Интерактивный доступ	0,06
3. Обеспечение хранения, ведения и поиска данных в сложных структурах	
4. Наличие у ПО одновременно нескольких характеристик по табл.Г4.1, приложение Г 4.1. 2 характеристики 4.2. 3 характеристики 4.3. Свыше 3–х характеристик	0,09

Поправочный коэффициент, учитывающий новизну разрабатываемого ПО ($К_н$) определяется на основе данных представленных в таблице 4.3 и составляет 1,1.

Нормативная трудоемкость ПО ($Т_н$) определяется на основе принятого в расчет $V_у$ и категории сложности, которая уточняется с учетом сложности

и новизны проекта и степени использования стандартных модулей при разработке.

В соответствии с этим, согласно укрупненным нормам времени на разработку ПО ((V_0)) и группы сложности: объем ПО (строки исходного кода, LOC) 9525, категория сложности ПО 2–я –$T_н = 273$, категория сложности ПО 40.

Следовательно, $T_о$ будет равно

$$T_0 = 273 \cdot 0.19 \cdot 0.8 \cdot 1 = 41{,}496 \text{ (чел./дн.)}$$

Численность исполнителей проекта ($Ч_р$) рассчитывается по формуле

$$Ч_р = \frac{T_о}{T_р \cdot Ф_{эф}} \quad (4.5)$$

Таблица 4.3 - Поправочные коэффициенты, учитывающие новизну ПО($К_н$).

Категория новизны	Степень новизны	Использование		Значение $К_н$
		На основе нового типа ПК	В среде новой ОС	
А	Принципиально новые ПО, не имеющие доступных аналогов	+ – + –	+ + – –	1,65 1,60 1,21 1,0
Б	ПО, являющиеся развитием определенного параметрического ряда ПО	+ – +	+ – –	1,1 0,10 0,8
В	ПО, являющиеся развитием определенного параметрического ряда ПО, разработанных для ранее освоенных типов конфигурации	–	–	0,5

где $Ф_{эф}$ − эффективный фонд времени работы одного работника в течение года (дни.);
$T_о$ − общая трудоемкость разработки проекта (чел./дни.);
$T_р$ − срок разработки проекта (лет).

Срок разработки проекта ($Т_р$) определяется по формуле

$$Т_р = \frac{Т_о}{Ч_р \cdot Ф_{эф}} \qquad (4.6)$$

где $Ч_р$ – плановое число разработчиков.

Эффективный фонд времени работы одного работника ($Ф_{эф}$) рассчитывается по формуле

$$Ф_{эф} = Д_г – Д_п – Д_в – Д_о, \qquad (4.7)$$

где $Д_г$ - количество дней в году;
$Д_п$ - количество праздничных дней в году;
$Д_в$ - количество выходных дней в году;
$Д_о$ - количество дней отпуска.

В соответствии с производственным календарем на 2016 год.
$Д_г$ – 366; $Д_п$ = 17; Дв – 104; $Д_о$ – 14,

$Ф_{эф}$ = 366 – 17 – 104 – 14 = 231 дней

Плановое число разработчиков $Ч_р$ = 1, следовательно, по формуле (4.6)
$Т_р$ = 41,496 / (1 · 231) = 0,17 года = 63 дней.

Таким образом, согласно произведенным расчетам и в соответствие с формулой (4.5)
Ч = 41,496 / (0,17 · 231) = 1 чел.

Основная заработная плата исполнителей на конкретное ПО рассчитывается по формуле

$$З_{о\,i} = \sum_{i=1}^{n} Т_{чi} \cdot Т_ч \cdot Ф_п \cdot К \qquad (4.8)$$

где n − количество исполнителей, занятых разработкой конкретного ПО;
$Т_{чi}$ − часовая тарифная ставка i–го исполнителя (тенге);
$Ф_п$ − плановый фонд рабочего времени i–го исполнителя (дней), 60 дня;
$Т_ч$ − количество часов работы в день (час), 8 часов;
К − коэффициент премирования, составляет 1,2.

По данным о специфике и сложности выполняемых функций составляется штатное расписание группы специалистов – исполнителей, участвующих в разработке ПО, с определением образнее специальности, квалификации и должности в таблице 4.4.

Таблица 4.4 - Сведения по работникам, задействованным в проекте

Специалист – Исполнитель	Количество, человек	Заработная плата в месяц, тенге
Программист	1	95000
Итого		95000

Часовая тарифная ставка рассчитывается путем деления месячной тарифной ставки на установленную при 40 - часовой недельной норме рабочего времени расчетную среднемесячную норму рабочего времени в часах ($Ф_р$) учитывая, что в месяце 21 рабочий день

$$Т_ч = \frac{Т_м}{Ф_р}, \qquad (4.9)$$

где $Т_ч$– часовая тарифная ставка (тенге);
$Т_м$– месячная тарифная ставка (тенге).
Таким образом
$Ф_р = 8 \cdot 21 = 168$ часов

$$Т_д = \frac{95\,000}{21} = 4523{,}8 \text{ тенге/час}$$

$$ФОТ = \frac{95\,000}{21} \cdot 60 \cdot 1{,}2 = 341\,999{,}9 \text{ тенге}$$

Социальный налог составляет 11% (ст. 358 п. 1 НК РК) от дохода работника, и рассчитывается по формуле

$$З_{сзі} = (ФОТ – ПО) \cdot 11\% \qquad (4.10)$$

где ПО – пенсионные отчисления, которые составляют 10% от ФОТ и социальным налогом не облагаются

$$ПО = ФОТ \cdot 10\% \qquad (4.11)$$

Таким образом
$ПО = 341\,999{,}9 \cdot 0{,}1 = 34\,199{,}99$ тенге

$З_{сзі} = (341\,999{,}9 – 34\,199{,}99) \cdot 0{,}11 = 33\,857{,}9$ тенге

Затрат на материалы определяются по формуле

$$M_i = (З_{осн.} \cdot Н_{мз})/ 100\% \qquad (4.12)$$

где $Н_{мз}$ − норма расхода материалов от основной заработной платы (3–5%).
M_i = 341 999,9 · 0,05 = 17 099,9 тенге

Амортизационные отчисления производятся по установленным нормам амортизации, выражаются, в процентах к балансовой стоимости оборудования и рассчитываются по формуле

$$A = \frac{С_{обор} \cdot Н_А \cdot N}{366 \cdot 100} \qquad (4.13)$$

где $Н_а$ – норма амортизации (25 %);
$С_{обор.}$– первоначальная стоимость оборудования;
N – Фактический срок эксплуатации оборудования, 60 дней;
Данные по стоимости оборудования представлены в таблице 4.5.

Таблица 4.5 - Стоимость оборудования одного ПК с периферией и ПО

Наименование	Цена без НДС, тенге
Монитор	30 800
Материнская плата	13 200
Процессор	35 200
Видеокарта	21 120
HDD	10 560
DVD–RW	5 720
CPU Cooler	4 840
Оперативная память	7 920
Клавиатура	1760
Мышь	1760
Принтер	7 040
Flash память	Бесплатно

Итого:	139 920

$C_{обор.}$ = 139 920 тенге
Тогда, согласно формуле (4.14)

$$A = \frac{139\,920 \cdot 25 \cdot 63}{366 \cdot 100} = 6021{,}15 \text{ тенге}$$

Расходы по статье «Прочие затраты» ($П_{зі}$) на конкретное ПО включает затраты на приобретение и подготовку специальной научно-технической информации и специальная литература. Определяются по нормативу, разрабатываемому в целом в организации, в процентах к основной заработной плате.

$$П_{зі} = З_{оі} \cdot Н_{пз}/100, \qquad (4.15)$$

где $Н_{пз}$– норматив прочих затрат в целом по организации (20 %),
Согласно формуле (4.16)
$П_{зі}$= 341 999,9 · 0,2 = 68 399,98 тенге

Затраты по статье «Накладные расходы» ($Р_{ні}$), связанные с необходимостью содержания аппарата управления, вспомогательных хозяйств и опытных(экспериментальных) производств, а также с расходами на общехозяйственные нужды ($Р_{ні}$), относятся на конкретное ПО по нормативу ($Н_{рн}$) в процентном отношении к основной заработной плате исполнителей. Норматив устанавливается в целом по организации

$$Р_{ні} = З_{оі} \cdot Н_{рн}/100\%, \qquad (4.16)$$

где $Р_{ні}$ – накладные расходы на конкретную ПО (тенге);
$Н_{рн}$ – норматив накладных расходов по организации (50%)
Согласно формуле (4.16)
$Р_{ні}$= 341 999,9 · 0,5 = 170 999,9 тенге

В соответствие с формулой (4.1) расчет полных затрат на разработку прикладного приложения составит
$С_{пі}$ = 341 999,9 + 33 857,9 + 17 099,9 + 6021,15 + 68 339,98 + 170 999,9 =

638 318,73 тенге

Сводные результаты расчета затрат на разработку ПО и их структура представлены в таблице 4.6 и на рисунке 4.1.

Таблица 4.6 - Затраты на разработку прикладного приложения для деловой игры рынок электроэнергии

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Социальный налог	$З_{сзі}$	33 857,9	5,3
Фонд оплаты труда	$З_{ФОТ}$	341 999,9	53,57
Материалы	$М_і$	17 099,9	2,67
Амортизация			
Прочие затраты	А	6021,15	0,94
Накладные расходы	$П_{зі}$	68 339,98	10
Итого:	$Р_{ні}$	170 999,9	26,7

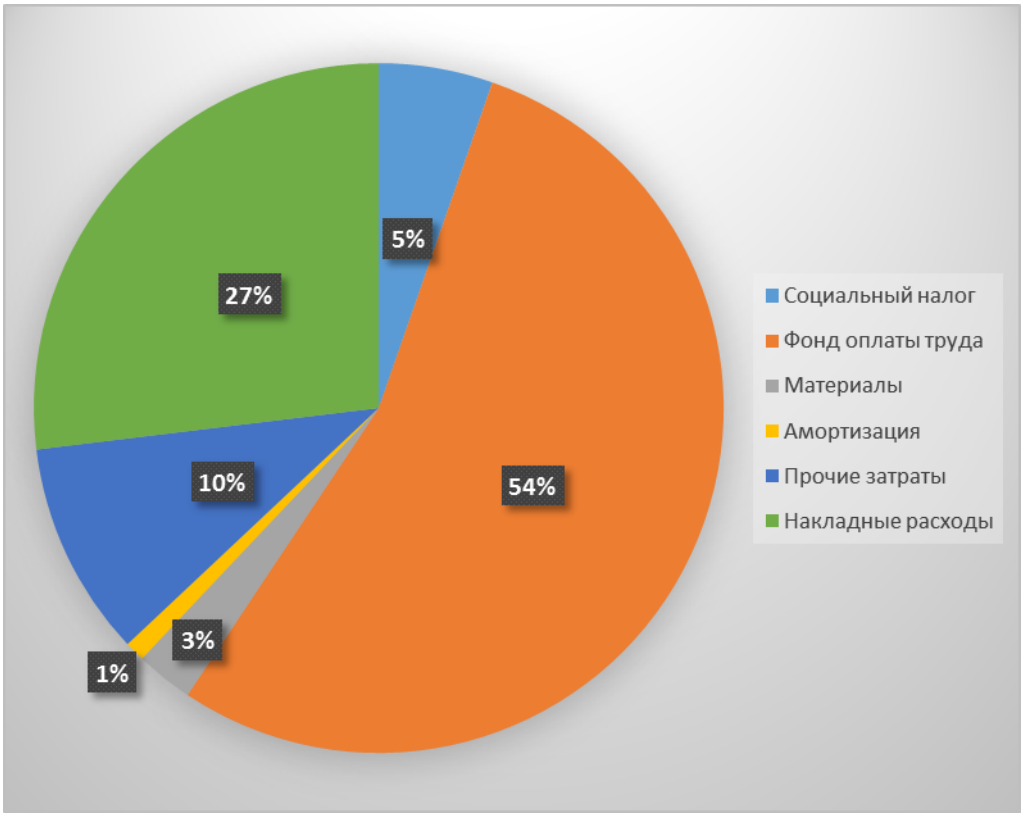

Рисунок 4.1 - Структура затрат на разработку прикладного приложения для деловой игры рынок электроэнергии

4.3 Расчет цены программного продукта

Расчет цены ПП, который разработан одной организаций по заказу другой и не предназначен для тиражирования, осуществляется по формуле

$$Ц_{ПП} = З_{РПР} + П_п + НДС \quad (4.17)$$

где $Ц_{ПП}$ – цена программного продукта, тенге;
$З_{РПР}$ – затраты на разработку проектного решения, в данном случае программного продукта, тенге;
$П_п$ – планируемая прибыль, тенге;
$НДС$ – налог на добавленную стоимость, тенге.

Планируемая прибыль рассчитывается по формуле

$$П_п = З_{РПР} \cdot R_{НПП} \quad (4.18)$$

где $R_{НПП}$ – нормативная рентабельность ПП, определяемая организацией (20%).

НДС, начисленный на ПП, определяется следующим образом

$$НДС = (З_{РПР} + П_п) \cdot k_{НДС} \quad (4.19)$$

где $k_{НДС}$ – ставка налога на добавленную стоимость.

Подставляем все значения в формулы (4.17) – (4.19) и получаем

$П_п = 638\,318{,}73 \cdot 0{,}2 = 127\,663{,}7$ тенге

Подставив данные в формуле (4.19) получаем

$НДС = (638\,318{,}73 + 127\,663{,}7) \cdot 0{,}12 = 91\,917{,}892$ тенге

Подставив данные в формуле (4.18) получаем

$Ц_{ПП} = 638\,318{,}73 + 127\,663{,}7 + 91\,917{,}892 = 857\,900{,}322$ тенге

4.4 Вывод

Разработка сайта является сложным и трудоемким процессом, требующих больших затрат интеллектуального труда. Стоимость разработки включает в себя следующие категории затрат: фонд оплаты труда,

отчисления на социальные нужды, амортизационные отчисления, затраты на электроэнергию. Стоимость разработки составила 857 900,322 тенге.

Наименьшую долю в общей сумме затрат на разработку ПП составляют прочие затраты 68 399,9 тенге, которая является определяющими при расчете затрат на оплату труда.

Цена реализации ПП составила 857 900,322 тенге.

Средняя рыночная цена на аналогичный ПП, с учетом на 30-50 пользователей колеблется в пределах от 300 400 до 950 600 тенге.

Средняя цена услуг на реализацию такого проекта на рынке составила 600 500 тенге. Эту цену мы получили, анализируя фонд оплаты труда различных веб-студий с учетом такого же количества работников и дней на реализацию проекта, как и у нас. Анализируя полученные расчеты, а точнее, полученную цену реализации проекта, равной 857 900,322 тенге, можно сделать вывод о том, что данный проект является экономически выгодным.

5 Безопасность жизнедеятельности

5.1 Анализ потенциально опасных и вредных факторов, воздействующих на обслуживающий персонал при эксплуатации технического оборудования

В данном дипломном проекте рассматривается разработка сайта научно-исследовательского центра «Интернет пространства и безопасность». Веб–сайт будет разрабатываться в офисе, в связи с чем необходимо учитывать вопросы создания оптимальных условий разработки, хорошее самочувствие, безопасность и сохранение здоровья.

Существует несколько вредных факторов, воздействующих на работников, занятых на работе с видео дисплейными терминалами (ВДТ) и персональными компьютерами (ПК):

1) воздействие электромагнитных полей (радиочастот), статического электричества;

2) неудовлетворительный микроклимат помещений;
3) недостаточная освещенность;
4) психоэмоциональное напряжение.

Без строгого учёта правил техники безопасности и производственной санитарии, неточного выполнения требований техники безопасности может привести к аварии, либо к профессиональным заболеваниям и производственному травматизму. Охрана труда обеспечивается системой законодательных актов, социально–экономических, организационных, технических, гигиенических и лечебно–профилактических мероприятий и средств, направленных на создание таких условий труда, при которых исключено воздействие на работающих опасных и вредных производственных факторов. Создание наиболее благоприятных, комфортных условий труда, улучшение охраны труда и техники безопасности, без сомнения, ведет к более высокой производительности труда, социальному развитию и повышению благосостояния.

Согласно ГОСТ 12.1.005–88. ССБТ «Оптимальные и допустимые нормы микроклимата, в зависимости от категории работ», работа людей в помещении относится к работе лёгкой тяжести (1а), так как управление оборудованием осуществляется дистанционно с помощью компьютеров.

С целью создания нормальных условий для работников предприятий
связи установлены нормы производственного микроклимата. В помещениях при работе с ЭВМ должны соблюдаться следующие климатические условия:

1) Холодный период года:
● оптимальная температура 22–24 ˚С, допустимая температура 18–26 ˚С;
● относительная влажность 40–60 %, допустимая влажность 75 %;
● скорость движения воздуха относительная и допустимая 0,1 м/с.
2) Тёплый период года:
● оптимальная температура 23–25 ˚С, допустимая температура 20–30 ˚С;
● относительная влажность 40–60 %, допустимая влажность 55 %;
● скорость движение воздуха относительная 0,1 м/с и допустимая 0,1–0,2 м/с.

5.2 Помещение, в котором ведется разработка

Рассмотрим помещение, в котором ведется разработка:
- тип помещения: офис;
- площадь помещения – 37 м^2;
- размеры рабочего помещения: длина 4 м, ширина 3 м, высота 3 м;
- остекление помещения – двойное(одно окно размером 2000×2000 мм);
- искусственное освещение – светильники: 2 светильника, в каждом по 2 люминесцентные лампы (ПВЛМ–1×40);

- внутренняя отделка стен – светлая;
- помещение по зрительным условиям работы относится к V разряду, т. к. наименьший объект различения от 1 до 5 мм;
- вид работы – разработка интернет приложения для размещения бесплатных объявлений;
- количество рабочих мест – 2;
- категория работ – легкая.
- расстояние от ПК – 80 см;
- количество работающих – 2.

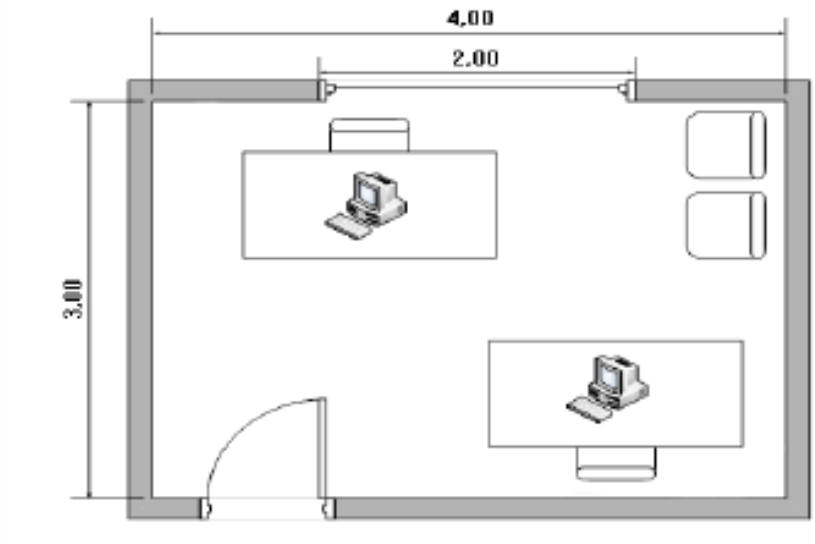

Рисунок 5.1 План помещения

5.3 Характеристики используемого оборудования

Персональный компьютер. Технические характеристики устройства:
– габариты: 1600×700×1050 мм (ноутбук + стол + кресло);
– электропитание: переменное напряжение 220–250 В, частотой 50 Гц, мощность 90 Вт;
– количество – 1 шт.
Модем:
– 4–х порт.коммутатором 10/100 Мбит/с;
– количество – 1 шт;
Установлен 1 оконный кондиционер Samsung SMG4590, характеристики:
– режим охлаждения – 2400 Вт;
– режим обогрева – 2400 Вт;
– поток – 404 $м^3$/ч;
– уровень шума 31Дб;
– габариты – 754×272×176 мм;

– масса – 39 кг;
– рассчитан на помещение площадью до 21 м^2.

5.4 Расчёт искусственного освещения методом коэффициента использования

Освещенность – световая величина, равная отношению светового потока, падающего на малый участок поверхности, к его площади. Единицей измерения освещённости в Международной системе единице (СИ) служит люкс (1 люкс = 1 люмену на квадратный метр), в СГС – фот (один фот = 10 000 люксов). В отличие от освещенности, выражение количества света, отражённого поверхностью, называется светимостью. Освещённость прямо пропорциональна силе света источника света. При удалении его от освещаемой поверхности её освещённость уменьшается обратно пропорционально квадрату расстояния. Когда лучи света падают наклонно к освещаемой поверхности, освещённость уменьшается пропорционально косинусу угла падения лучей.

При проектировании искусственного освещения помещения необходимо определить площадь световых проемов, обеспечивающих нормированное значение в соответствии с требованиями.

Разряд зрительной работы – V.

Нормируемая освещённость – 400 лк.

Используем систему общего освещения с люминесцентными лампами ЛБ с параметрами, указанными в таблице 5.1.

Т а б л и ц а 5.1 – Технические характеристики газоразрядных ламп ЛБ

Номинальная мощность, Вт	Номинальный поток ламп типа ЛБ, лм	Размеры ламп, мм	
		Диаметр	Длина по штырькам
40	3120	40	1213,6

Вычислим высоту подвеса светильника над рабочей поверхностью

$$H = h - h_p - h_c \qquad (5.4)$$

где h_c– расстояние от светильника до перекрытия, $h_c = 0{,}05$ м;

$h_р$– высота рабочей поверхности над полом, $h_р$ = 0,7 м;
h – высота помещения, h = 3 м.
H = 3-0.7-0.05 = 2.25 м

Наиболее выгодное расстояние между светильниками определяется как

$$L = \lambda \cdot H \tag{5.5}$$

где $\lambda = 1{,}2 \div 1{,}4$
L = $1{,}2 \cdot 2{,}25 = 2{,}7$м

Расстояние от стены до ближайшего светильника, когда работа у стены не проводится, определяем по формуле

$$l_1 = (0{,}4 \div 0{,}5) \cdot L \tag{5.6}$$

$l_1 = 0{,}4 \cdot 2{,}7 = 1{,}08$м

Определяем индекс помещения

$$i = \frac{a \cdot b}{H \cdot (a+b)} \tag{5.7}$$

$$i = \frac{4{,}3}{2{,}25 \cdot (4+3)} = 0{,}76$$

Коэффициенты отражения от потолка стен и пола соответственно равны
$\rho_{пот}$ = 70 %
$\rho_{ст}$ = 50 %
$\rho_{пол}$ = 30 %

Коэффициент использования в данном случае равен η=40 %, коэффициент запаса равен $к_З$ = 1,2.

Определим количество люминесцентных ламп по формуле

$$N = \frac{E \cdot к_з \cdot S_{ОС} \cdot Z}{n \cdot Ф_Л \cdot \eta} \tag{5.8}$$

где $S_{ОС}$– площадь помещения;
$k_З$– коэффициент запаса, $k_З$ =1,2;
E – заданная минимальная освещённость, E = 400 лк.;
Z – коэффициент неравномерности освещения, Z = 1,1;
n – количество ламп в светильнике;
Фл – световой поток выбранной лампы, Фл = 3120 лм;
η – коэффициент использования, η = 40 %.

$$N = \frac{400 \cdot 1{,}2 \cdot 12 \cdot 1{,}1}{2 \cdot 3120 \cdot 0{,}4} = 2{,}5 \approx 3$$

Всего для создания нормируемой освещенности 400 лк необходимо 3 светильника серии ЛБ с 2 лампами в светильнике, итого 6 люминесцентных ламп, мощность каждой лампы должна быть не меньше 40 Вт, что не соответствует действительности, а значит имеющегося в

наличии освещения не достаточно для соответствия санитарным нормам. Так как в помещении установлено всего 2 светильника, а это нарушение санитарных норм, необходимо установить еще один светильник. В результате было реконструировано искусственное освещение, в итоге насчитывается 3 светильника серии ЛБ с 2 лампами в светильнике, что соответствует санитарным нормам.

5.5 Расчет искусственного освещения точечным методом

Сделаем проверочный расчет точечным методом. Точечный метод расчета используется при применении точечных излучателях (лампы накапливания и ДРЛ). Попробуем применить этот метод и для люминесцентной лампы выбранной нами в качестве источника света в помещении.

Световой поток лампы в каждом светильнике определяется по формуле:

Разряд зрительной работы – V.

Нормируемая освещённость – 400 лк.

Высота подвеса светильников над освещаемой поверхностью H = 2,25 м, коэффициент запаса равен $\kappa_3 = 1{,}2$ (Табл.5.2). Схема освещённости представлена на рисунке 5.2.

Таблица 5.2-Светораспределение светильников

Тип св-ка	Сила света Iα, кд в направлении угла α										
	0	5	15	25	35	45	55	65	75	85	90
ПВЛМ-1×40	139	135	132	115	104	84	63	44	22	6	0

Намечаем контрольную точку А. Для нее определяем суммарную условную освещенность всех светильников следующим образом.

Находим проекцию расстояния на потолок от точки А до светильника–d.

Далее определяем угол между потолком и прямой d. По этому углу находим условную освещенность.

Пусть контрольная точка будет находится под лампой №2. Обозначим d_i – расстояние от контрольной точки до i–ой лампы

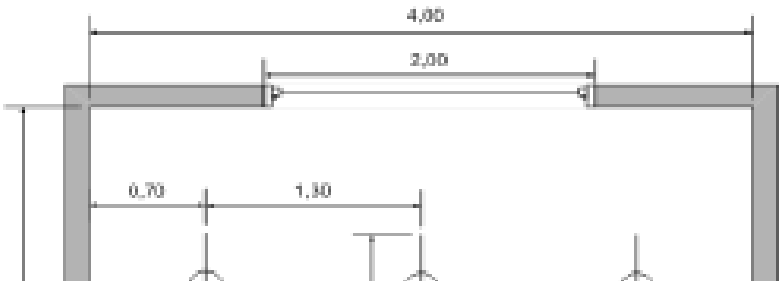

Рисунок 5.2 – Схема освещения

$$\alpha = \text{arcg}(\frac{d_i}{H}), \quad (5.9)$$

$$e_i = \frac{I_\alpha \cdot cos^2(\alpha)}{H^2} \quad (5.10)$$

1-я лампа $d_2 = 0$ м
$\alpha = 0^0$;
$e_i = \frac{139 \cdot 2 \cdot cos^2(0)}{2,25} = 54,9$ лк
1,3-я лампы d1 = d3 = 1,3 м;
$\alpha = 30^0$;
$e_i = \frac{110 \cdot 2 \cdot cos^2(30)}{2,25^2} = 28,22$ лк
Суммарная условная освещенность равна
$\Sigma^E = 54,9 + 2 \cdot 28,22 = 111,34$ лк

Суммарная освещенность равна

$$E_{АГ} = \frac{1,1 \cdot 2 \cdot 3120}{1000 \cdot 1,2} \cdot 111,34 \approx 640 \text{ лк}$$

В результате проверки мы убедились, что искусственного освещения вполне достаточно.

5.6 Требования пожарной безопасности к содержания зданий, сооружений и помещений

Для выполнения противопожарный требований здесь необходимо выполнять следующие правила:

- для всех производственных и складских помещений должна быть определена в соответствии с НПБ 105-03 категория взрывопожарной и пожарной опасности, а также класс зоны по Правилам устройства электроустановок, которые надлежит обозначить на дверях помещений;
- около оборудования, имеющего повышенную пожарную опасность, следует вывешивать стандартные знаки безопасности;
- противопожарные системы и установки (против дымная защита, средства пожарной автоматики, системы противопожарного водоснабжения, противопожарные двери, клапаны, другие защитные устройства в противопожарных стенах и перекрытиях и т.п.) помещений, зданий и сооружений должны постоянно содержаться в исправном рабочем состоянии;
- устройства для само закрывания дверей должны находиться в исправном состоянии. Не допускается устанавливать какие-либо приспособления, препятствующие нормальному закрыванию противопожарных или против дымных дверей (устройств);
- наружные пожарные лестницы и ограждения на крышах (покрытиях) зданий и сооружений должны содержаться в исправном состоянии и не реже одного раза в пять лет подвергаться эксплуатационным испытаниям;
- в помещениях с одним эвакуационным выходом одновременное пребывание 50 и более человек не допускается. В зданиях с массовым пребыванием людей на случай отключения электроэнергии у обслуживающего персонала должны быть электрические фонари. Количество фонарей определяется руководителем исходя из особенностей объекта, количества дежурного персонала, количества людей в здании, но не менее одного на каждого работника дежурного персонала;
- двери чердачных помещений, а также технических этажей и подвалов, в которых по условиям технологии не требуется постоянного пребывания людей, должны быть закрыты на замок. На дверях указанных помещений должна быть информация о месте хранения ключей. Окна чердаков, технических этажей и подвалов должны быть остеклены и постоянно закрыты;
- во всех производственных, административных, складских и вспомогательных помещениях на видных местах должны быть вывешены таблички с указанием номера телефона вызова пожарной охраны.

В зданиях, сооружениях организаций запрещается:

- хранение и применение в подвалах и цокольных этажах ЛВЖ и ГЖ, пороха, взрывчатых веществ, баллонов с газами, товаров в аэрозольной упаковке, целлулоида и других взрывопожароопасных веществ и материалов, кроме случаев, оговоренных в действующих нормативных документах;
- использовать чердаки, технические этажи, вент камеры и другие

технические помещения для организации производственных участков, мастерских, а также хранения продукции, оборудования, мебели и других предметов;

- размещать в лифтовых холлах кладовые, киоски, ларьки и т.п.;
- устраивать склады горючих материалов и мастерские, размещать иные хозяйственные помещения в подвалах и цокольных этажах, если вход в них не изолирован от общих лестничных клеток;
- снимать предусмотренные проектом двери эвакуационных выходов из поэтажных коридоров, холлов, фойе, тамбуров и лестничных клеток, другие двери, препятствующие распространению опасных факторов пожара на путях эвакуации. Производить изменения объемно - планировочных решений, в результате которых ухудшаются условия безопасной эвакуации людей, ограничивается доступ к огнетушителям, пожарным кранам и другим средствам пожарной безопасности или уменьшается зона действия автоматических систем противопожарной защиты;
- загромождать мебелью, оборудованием и другими предметами двери, люки на балконах и лоджиях, переходы в смежные секции и выходы на наружные эвакуационные лестницы;
- проводить уборку помещений и стирку одежды с применением бензина, керосина и других ЛВЖ и ГЖ, а также производить отогревание замерзших труб паяльными лампами и другими способами с применением открытого огня;
- оставлять неубранным промасленный обтирочный материал;
- устанавливать глухие решетки на окнах и приямках у окон подвалов, за исключением случаев, специально оговоренных в нормах и правилах, утвержденных в установленном порядке;
- остеклять балконы, лоджии и галереи, ведущие к незадымляемым лестничным клеткам;
- устраивать в лестничных клетках и поэтажных коридорах кладовые (чуланы), а также хранить под лестничными маршами и на лестничных площадках вещи, мебель и другие горючие материалы. Под лестничными маршами в первом и цокольном этажах допускается устройство только помещений для узлов управления центрального отопления, водомерных узлов и электрощитов, выгороженных перегородками из негорючих материалов;
- устанавливать дополнительные двери или изменять направление открывания дверей (в отступлении от проекта) из квартир в общий коридор (на площадку лестничной клетки), если это препятствует свободной эвакуации людей или ухудшает условия эвакуации из соседних квартир;

5.7 Требования к организации и оборудованию рабочих мест

1. Рабочие места по отношению к световым проемам должны располагаться так, чтобы естественный свет падал сбоку, преимущественно слева.

2. Расстояние между рабочими столами с видеомониторами (в направлении тыла поверхности одного монитора и экрана другого) должно быть не менее 2,0 м, а расстояние между боковыми поверхностями видеомониторов не менее 1,2 м.

3. Оконные проемы должны быть оборудованы регулируемыми устройствами: жалюзи, занавесей, внешних козырьков и др.

4. Конструкция рабочего стола должна обеспечивать оптимальное размещение на рабочей поверхности оборудования.

5. Конструкция рабочего стула должна обеспечивать поддержание рациональной рабочей позы при работе на ПЭВМ, позволять изменять позу с целью снижения статического развития утомления.

6. Экран видеомонитора должен находиться от глаз пользователя на расстоянии 600-700 мм, но не ближе 500 мм с учетом размеров знаков и символов.

7. Рабочее место должно быть оснащено пюпитром для документов.

8. Высота края стола, обращенного к работающему с ПК, высота пространства для ног должна соответствовать росту работающего.

9. Уровень глаз при вертикально расположенном экране должен приходиться на центре или 2/3 высоты экрана. Линия взора должна быть перпендикулярна экрану и оптимальное ее отклонение от перпендикуляра, проходящего через центр экрана в вертикальной плоскости, не должно превышать 5 градусов, допустимое 10 градусов.

Вывод

В этом разделе был произведен анализ условий труда в данном офисном помещении, в частности, расчет искусственного освещения. Искусственное освещение рабочего места было недостаточным, что послужило поводом на реконструкцию освещения офиса. В итоге был добавлен еще один светильник. Таким образом, система искусственного освещения включает в себя 3 светильника по 2 лампы со световым потоком излучения 3120 лм каждая, поэтому в этом помещении можно работать и в темное время суток.

ЗАКЛЮЧЕНИЕ

Целью работы является создание сайта, освещающего работу научно-исследовательского центра «Интернет пространство и безопасность». В работе показаны разносторонние материалы показывающий актуальность

создания научного центра. Был разработан сетевой СУБД, таблицы стилей CSS, язык программирования JavaScript, создание графики с помощью Adobe Photoshop, а сам узел создавался в программе WordPress 2.0. Также рассмотрены языки разметки гипертекстовых документов HTML, его основные функции и параметры. Сегодня применение HTML практикуется во всех без исключения электронных документов, независимо от тематики, величины и коммерческой направленности Интернет проекта.

Так же в дипломной работе рассмотрены актуальные вопросы разработки и создания современного Web-сайта.

При этом были решены следующие частные задачи:

- ознакомлены с современными Интернет-технологиями и их использование в настоящей разработке;

- изучение программного инструментария, применяемого для разработки и создания Web-сайтов;

- выявление и учет методов и способов представления на Web-страницах различных видов информации, не препятствующих их доступности;

- ознакомление с основными правилами и рекомендациями по разработке и созданию Web-сайтов и неукоснительное следование им на практике;

- определение структуры Web-страниц;

- выбор стратегии разработки и создания Web-сайта.

В результате проведенных работ на базе выбранных технологий был создан прототип современного Web-сайта.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Э. Крамер. HTML – Наглядный курс Web-дизайна. М.-Спб.Киев: Диалектика,2001 г.
2. Пол Макфедрис. Создание Web-страниц. М.: АСТ Астрель,2005 г.
3. Интернет-ресурсы.
4. Ганчаров А. Самоучитель HTML. Питер 2000 г.
5. Денисов Internet Explorer 5 Справочник 2011 г.
6. Хоумер А. Dynamic HTML Справочник 2011 г.
7. Петюшкин А.В., HTML. Экспресс-курс. – СПб.: БХВ – Петербург, 2003 г.
8. Кингсли-Хью Э., JavaScript: учебный курс. – СПб.: Питер, 2010 г.
9. http://www.robotland/ru/
10. http://lanserv1.kemsu.ru/oruch3.html

11. http://www.helloword.ru/texts/comp/web/html/html3/lab 1.htm
12. http://www.sao.ru/racs/reques/diplom.html
13. http://mail 1.kemsu.ru/obuch3.html
14. Хоумер А. Dynamic HTML: справочник программиста. – СПб.: ПитерКом, 2006 г.
15. Глушаков С.В., Жакин И.А., Хачиров Т.С. Программирование Web-страниц. – Харьков: Фолио, 2012 г.

ПРИЛОЖЕНИЕ

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251"/>
<title>E-Canteen</title>
<link href="style.css" rel="stylesheet" type="text/css">
</head>
<body>
```

```
<!main container start-->
<div id="mainContainer">
<!--header start-->
<div id-"header">
<a href="index.html" title=e Grapes"><imgsrc="images/logo2.gif" alt="e Grapes" border="0" /></a>
</ul>
<li><a href="index.html" title="Home">Home</a></li>
<li><a href="dishes.html" title="Dishes">Dishes</a></li>
<li><a href="beverages.html" title="Beverages">Beverages</a></li>
<li><a href="cart.html" title="Cart">Cart</a></li>
<li><a href="aboutus.html" title="About us">About us</a></li>
<li><a href="contact.php" title="Contact">Contact</a></li>
</ul>
</div>
<!--header end-->
<!--body stars-->
<div id="body">
```

Продолжение приложения

```
<!--left pannel start-->
<div id="left">
<span class="leftTop"></span>
<form method="post" action="login.php" name="login">
<h2 class="login">Login area</h2><br class="spacer"/>
<label>Your name</label><br class="spacer"/>
<input name="lname" type="text" id="lname"/>
<br class="spacer"/>
<label>Your password</label><br class="spacer"/>
<input name="password" type="password"id="password"/><br class="spacer"/>
<p>
<a href="#" title="Fofget your password? - click here">forget password</a><br class="spacer"/>
```

Want to register<br class="spacer"/>

<input name="login" type="image" class="loginBtn" id="login" title="Login" src="images/login_btn.gif"/>

<br class="spacer"/></form>

<form method="post" actoin="#"name="login">

<h2 class="contact">Contact</h2><br class="spacer"/>

<label>Name</label><br class="spacer"/>

<input name="cname"type="text" id="cname"/>

<br class="spacer"/>

<label>Address</label><br class="spacer"/>

<input name="address"type=text"id="address"/><br class="spacer"/>

<label>Email id</label><br class="spacer"/>

<input name="email" type="text"id="email"/><br class="specer"/>

Продолжение приложения

<label>Comments</label><br class="spacer"/>

<textarea name="comments" cols="15" rows="5" id="comments"></textarea><br class="spacer"/>

<input name="" type="image" src="images/submit_btn.gif" title="Submit" class="submit"/>

<input name="" type="reset" value="reset"title="Reset"class="reset"/>

<br class="spacer"/></form>

<!--left pannel end-->

<!--midle start-->

<h2 class="welcom">Welcome..</h2>

We are glad to see You here!This is the online site of E-Canteen,a chain of canteens. We provide very useful options to every Guest in our cafes.

If You are interested to spend Your time effectively, You ae in the right place!

<!-- more -->

<imgsrc="images/best.jpg" alt="/Our best Works" width="260" height="125" border="0" title="Our Best Dishes">

<!--<h2 class="event">Events..</h2>-->

<u>From Soups<u>,
Consomme Celestine with pumpkin just only 2$ for portion!

Eat now!

<u>From Main Course</u>,
Roast chicken breast with asparagus tips just only 5.10$ for portion!

Продолжение приложения

Eat now!<small>

<u>From Desserts</u>,
Blinis of banana with caramel ice-cream and vanilla sauce just only 2.30$ for portion!

Eat now!<small>

<!--Шаблон сайта скачен с http://themebot.ru/html-shablony.
Если желаете посмотреть всю коллекцию, посетите http://themebot.ru/html-shablony - более 100 шаблонов. </small>

-->

more

<!--midle end-->

<!--right pannel start-->

<!--rignt pannel end-->

<br class="spacer"/>

<!--body end-->

<!--main container start-->

<!footer start-->

```
<div id="footer">
<div class="footer">
<ul>
<li><a href="index.html" title="Home">Home</a></li>
<li><a href="dishes.html" title="Dishes">Dishes</a></li>
```

Продолжение приложения

```
<li><a href="beverages.html" title="Beverages">Beverages</a></li>
<li><a href="cart.html" title="Cart">Cart</a></li>
<li><a href="aboutus.html="About us">About us</a></li>
<li><a href="contact.php" title="Contact"Contact</a></li>
</ul>
</div>
<!--footer end-->
</body>
</html>
<?php get_header();?>
<div class="art-contentLayout">
<div class="art-content">

<?php if (have_posts()): while (have_posts()):the_post();?>
<div class="art-Post">
  <div class="atr-Post-tl"></div>
  <div class="atr-Post-tr"></div>
  <div class="atr-Post-bl"></div>
  <div class="atr-Post-br"></div>
  <div class="atr-Post-tc"></div>
  <div class="atr-Post-bc"></div>
  <div class="atr-Post-cl"></div>
  <div class="atr-Post-cr"></div>
  <div class="atr-Post-cc"></div>
  <div class="atr-Post-body">
<div class="atr-Post-inner art-article">
```

<h2 class="art-PostHeaderIcon-wrapper">

Продолжение приложения

<a href="<?php the_permalink()?>"rel="bookmark"title="<?php printf(__('Permanent Link to %s', 'kubrick'),the_title_attribute('echo=0'));?>">

<?php the_title();?>

</h2>

<?php $icons=array();?>

<?php if (!is_page()):?><?php ob_start();?><?php the_time(__('F jS, Y', 'kubrick'))?>

<?php $icons[] = ob_get_clean(); ?><?php endif; ?><?php if(!is_page()): ?><?php ob_start(); ?><?php_e('Author', 'kubrick'); ?>: <a href="#" title="<?php_e('Author', 'kubrick'); ?>"><?php the_authour()?>

<?php $icons[] = ob_get_clean(); ?><?php endif; ?><?php if (current_user_can('edit_post' $post->ID)): ?><?php ob_start(); ?><?php edit_post_link(__('Edit', 'kubrick'), ''); ?>

<?php $ocons[] = ob_get_clean() ?><?php endif; ?><?php if(0!=count($icons)):?>

<div class-"art-PostHeaderIcons art-metadata-icons">

<?php echo implode('|', $icons); ?>

<?php endif; ?>

<?php if (is_search()) the_excerpt(); else the_content(__('Read the rest of this entry »','kubrick')); ?>

<?php ob_start(); ?>

<?php $icons = array(); ?>

<?php if (!is_page()): ?><?php ob_start(); ?><?php printf(__('Posted in %s', 'kubrick'), get_the_category_list(',')); ?>

Продолжение приложения

<?php $icons[] = ob_get_clean(); ?><?php endif; ?><?php if (!is_page() && get_the_tags()): ?><?php ob_start(); ?><?php the_tags(__('Tags:', 'kubrick'). ", ', ',"); ?>

```
<?php $icons[] = ob_get_clean(); ?><?php endif;  ?><?php if (!is_page() && !is_single()): ?><?php ob_start(); ?><?php comments_popup_link(__('No Comments &#187;','kubrick'),__('1 Comment &#187;','kubrick'),__('% Comments &#187;','kubrick'),'',__('Comments Closed','kubrick'));?>

<?php $icons[] = ob_get_clean(); ?><?php endif; ?><?php if(0!=count($icons)):?>

<div class="art-PostFooterIcons art-metadata-icons">

<?php echo implode('|',$icons);?>


</div>

<?php endif;?>

<?php $metadataContent = ob_get_clean();?>

<?php if(trim($metadataContent)!=''):?>

<div class="art-PostMetadataFooter">

<?php echo $metadataContent;?>


</div>

<?php endif;?>


</div>


  </div>

</div>


<?php endwhile; endif;?>
```

Продолжение приложения

```
</div>

<?php include (TEMPLATEPATH. '/sidebar1.php');?>

</div>

<div class="cleared"></div>

<?php get_footer();?>

<?php get_header();?>

<div class="art-contenrLayout">
```

```
<div class="art-content">
<?php if (have_posts()):?>
<div class="art-Post">
<div class="atr-Post-tl"></div>
<div class="atr-Post-tr"></div>
<div class="atr-Post-bl"></div>
<div class="atr-Post-br"></div>
<div class="atr-Post-tc"></div>
<div class="atr-Post-bc"></div>
<div class="atr-Post-cl"></div>
<div class="atr-Post-cr"></div>
<div class="atr-Post-cc"></div>
<div class="atr-Post-body">
<div class="art-Post-inner art-article">
<div class="art-PostContent">
<h2><?php_e('Search Results', 'kubrick');?></h2>
<?php
$prev_link=get_previous_posts_link(__('Newer Entries &raquo;','kubrick'));
$next_link=get_next_posts_link(__('&laquo;Older Entries','kubrick'));
?>
```

Продолжение приложения

```
<?php if ($prev_link || $next_link):?>
<div class="navigation">
<div class="alignleft"><?php echo $next_link;?></div>
<div class="alignleft"><?php echo $prev_link;?></div>
</div>
<?php endif;?>
</div>
<div class="cleared"></div>
</div>
  </div>
</div>
<?php while(have_posts()):the_post();?>
```

<h2 class="art-PostHeaderIcon-wrapper">

<a href="<?php the_permalink()?>"rel="bookmark" title="<?php printf(__('Permanent Link to %s', 'kubrick'),the_title_attribute('echo=0'));?>">

Продолжение приложения

<?php the_title();?>

</h2>

<?php $icons=array();?>

<?php if (!is_page()):?><?php ob_start();?><?php the_time(__('F jS, Y', 'kubrick'))?>

<?php $icons[] = ob_get_clean();?><?php endif;?><?php if (!is_page()):?><?php ob_start();?><?php_e('Author','kubrick');?>:<a href="#"title="<?php_e('Author','kubrick');?>"><?php the_author()?>

<?php $icons[] = ob_get_clean();?><?php endif;?><?php if ()current_user_can('edit_post', $post->ID)):?><?php ob_start();?><?php edit_post_link(__('Edit',"kubrick"),");?>

<?php $icons[] = ob_get_clean();?><?php endif;?><?php if (0!=count($icons)):?>

<>div class="art-PostHeaderIcons art-metadata=icons">

<?php echo implode('|',$icons);?>

<?php endif;?>

<?php if (is_search()) the_excerpt(); else the_content(__('Read the rest of this entry &raqui;','kubrick'));?>

<?php ob_start();?>

<?php $icons = array();

<?php if (!is_page()): ?><?php ob_start(); ?><?php printf(__('Poster in %s', 'kubrick'),get_the_category_list(',')); ?>

<?php $icons[] = ob_get_clean(); ?><?php endif; ?><?php if (!is_page() && get_the_tags()): ?><?php ob_start(); ?><?php the_tags(__('Tags:','kubrick').",',','); ?>

Продолжение приложения

<?php $icons[] = ob_get_clean(); ?><?php endif; ?><?php if (!is_page() &&!is_single()): ?><?php ob_start; ?><?pho comments_popup_link(__('No Comments »','kubrick'),__('1 Comment »','kubrick'),__('% Comments »','kubrick'),'',__('Comments Closed','kubrick')); ?>

<?php $icons[] = ob_get_clean(); ?><?php endif;?><?php if(0!=count($icons)):?>

<?php echo implode('|'$icons);?>

<?php endif;?>

<?php $metadataContent = ob_get_clean(); ?>

<?php if (trim($metadataContent)!=''): ?>

<?php echo $metadataContent; ?>

<?php endif; ?>

<?php endwhile; ?>

<?php if ($prev_link||$next_link): ?>

```
<div class="atr-Post-bl"></div>
<div class="atr-Post-br"></div>
<div class="atr-Post-tc"></div>
<div class="atr-Post-bc"></div>
<div class="atr-Post-cl"></div>
<div class="atr-Post-cr"></div>
```

Продолжение приложения

```
<div class="atr-Post-cc"></div>
<div class="atr-Post-body">
```