бюджетное образовательное учреждение

среднего профессионального образования Вологодской области

«Вологодский колледж технологии и дизайна»

УТВЕРЖДЕНО

приказом директора

БПОУ ВО «Вологодский колледж технологии и дизайна»

от 31.08.2022 № 580

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРАКТИЧЕСКИМ ЗАНЯТИЯМ

ПО УЧЕБНОМУ ПРЕДМЕТУ

**ОП.11 WEB-ДИЗАЙН**

ПО СПЕЦИАЛЬНОСТИ

**54.02.01 ДИЗАЙН (ПО ОТРАСЛЯМ)**

Вологда

2022

Методические рекомендации составлены в соответствии с ФГОС СПО по специальности 54.02.01 Дизайн (по отраслям) и рабочей программой учебной дисциплины ОП.11 Web-дизайн

Организация-разработчик: БПОУ ВО «Вологодский колледж технологии и дизайна»

Разработчик: Комиссарова Н.В.*,* преподаватель БПОУ ВО «Вологодский колледж технологии и дизайна»

Рассмотрено и рекомендовано к утверждению на заседании предметной цикловой комиссии БПОУ ВО «Вологодский колледж технологии и дизайна», протокол № 1 от 31.08.2022 г.

**ВВЕДЕНИЕ**

**Практические занятия** - одна из важнейших форм контроля самостоятельной работой обучающихся над учебным материалом, качеством его усвоения. Готовясь к практическим занятиям, обучающиеся должны изучить рекомендованную литературу: первоисточники, соответствующие разделы учебников, учебных пособий, конспекты лекций и т.д.

**Цель практических занятий –** формирование практических умений: выполнение определённых действий, операций, необходимых в последующей профессиональной или учебной деятельности. В связи с этим содержанием практических занятий является решение задач, выполнение вычислений, расчётов, работа с литературой, работа с лекциями, справочниками, инструкциями. Выполнению практических занятий может предшествовать проверка знаний обучающихся, их теоретической готовности к выполнению заданий.

Целями настоящих методических материалов является оказание содействия обучающимся в успешном освоении дисциплины ОП.09 Web-дизайн, необходимых теоретических основ представления о возможностях современного WEB дизайна, выработке навыков необходимых для самостоятельного проектирования и создания web сайтов с помощью компьютерных программ, в соответствии с общими целями основной образовательной программы.

1. **ЯЗЫК HTML**

HTML (Hypertext Markup Language – язык разметки гипертекста), стандартный язык разметки документов для WWW (World Wide Web – Всемирная паутина). HTML-документ интерпретируется браузерами и отображается в виде веб-страницы в удобной для просмотра визуальной форме.

HTML – «авторский» язык. Первая его версия была предложена в 1989 году Тимом Бернерс-Ли (Tim Berners-Lee) в рамках его проекта «Системы гипертекстовых документов», которая должна была использоваться внутри Европейского Центра ядерных исследований (CERN). В 1990 году эта система была названа World Wide Web. Основными составляющими Всемирной паутины стали язык разметки гипертекста (HTML), протокол передачи гипертекста (HTTP), универсальный локатор ресурса (URL).

Активно поддерживался этот язык компанией NSCA (Национальный центр суперкомпьютерных приложений), где был реализован один из первых Web-браузеров – Mosaic.

HTML является приложением SGML (Standard Generelized Markup Language - Cтандартный обобщённый язык разметки) и соответствует международному стандарту ISO 8879.

**Версии HTML-языка:**

HTML 2.0 – 1994; HTML 3.2 – 1996; HTML 4.0 – 1997; HTML 4.1 – 1999 HTML 4.0 (Specification - http://www.w3.org/TR/html401/)

HTML 5 - в разработке. Конец разработки запланирован на 2014 год. Стандарт HTML и другие стандарты для Web разработаны под

руководством World Wide Consortium (консорциум World Wide Web) – W3C. Стандарты, спецификации и проекты новых предложений можно найти на сайте http://www.3w.org

Другим источником информации о тегах HTML является HTML Compendium (краткое руководство по HTML), созданный Ron Woodal. Компендиум HTML находится по адресу http://www.htmlcompendium.org.

HTML определяет синтакcиc и размещение специальных инструкций (тегов) веб-документа, которые не выводятся на экран, но указывают браузеру, как отображать содержимое документа.

* + основном, HTML-документ является простым текстовым файлом, который содержит текст (содержимое документа) и специальные команды, называемые тегами, с инструкциями о структуре, внешнем виде
* функции содержимого.

Для обозначения тегов используется команда <тег>. Имя тега, как правило, представляет собой аббревиатуру его функции, что облегчает его запоминание.

Теги бывают двух видов: парные, которые также называют контейнером и одиночные.

Контейнер имеет начальный (открывающий) и конечный (закрывающий) тег. Текст, находящийся между тегами, будет выполнять содержащиеся в них инструкции. Конечный тег имеет то же имя, что и начальный, но перед ним стоит слэш (/), обозначаемый как </тег>.

Одиночные теги не имеют закрывающих тегов, так как они используются для размещения отдельных элементов на странице. Например: тег изображения - <img>, разрыв строки - <br>, горизонтальная линия - <hr> и т.д.

Теги нечувствительны к регистру, поэтому запись <P> и <p> эквивалентна.

**1.1 HTML-редакторы**

HTML-редакторы можно разделить на 4 типа:

* «визуальные» (WYSIWYG);
* полу-WYSIWYG;
* не-WYSIWYG;
* простые текстовые редакторы (н-р Блокнот).

Не-WYSIWYG редакторы представляют собой те же простые текстовые редакторы, снабженные функциями подсветки синтаксиса HTML. Пользоваться такими редакторами довольно удобно, но они требуют хорошего знания языка для успешной работы.

Например: WEB-ED, Notepad, UltraEdit

Типичным (и наиболее известным) полу-WYSIWYG редактором является HomeSite. Встроены специальные «мастеры», помогающие написать те или иные теги, инструменты и справочники по HTML. Плюс таких редакторов в том, что они позволяют писать «чистый» код.

Например: HomeSite, HotDog, FirstPage, AceHTML Pro

WYSIWYG редакторы относят к системам верстки. Знаний HTML практически не требуется, всю работу по генерации HTML-кода осуществляет сама программа. Однако, код полученный с помощью таких редакторов не всегда бывает оптимальным.

Например: Macromedia Dreamweaver, Microsoft FrontPage, Adobe GoLive, NetObjects Fusion

**1.2 Структурные теги HTML**

Код программы HTML документа разделяется структурно на две основные части: <head> и <body>. В head содержится метаинформация о документе, например, язык, на котором пишется текст, версия языка HTML, ключевые слова, общий заголовок HTML-документа. В body находится само содержимое документа (то, что выводится в окне браузера).

Пример демонстрирует теги, образующие основной структурный скелет документа HTML:

<HTML>

<HEAD>

<TITLE> Название документа </TITLE>

</HEAD>

<BODY> Содержимое документа

</BODY>

</HTML>

Рассмотрим теги управляющие структурой документа, а также предоставляющие информацию о документе, управляющие его внешним видом или функционированием на внешнем уровне.

**Тег-контейнер HTML**

Тег помещают в начало и конец документа, обозначает начало и завершение создания HTML-документа.

**Тег-контейнер HEAD**

Тег определяет заголовочную часть документа. Атрибуты отсутствуют. Тег используется в качестве контейнера для других тегов, таких как <meta>, <title>, <link>, <script>.

**Тег-контейнер TITLE**

Определяет название документа.

**Тег-контейнер BODY**

Тег определяет начало и конец основной части документа. Атрибуты этого тега оказывают влияние на весь документ.

**Основные атрибуты тега <body>:**

* **background = url** -определяет изображение,которое будет служитьфоном для страницы
* **bgcolor = “#rrggbb” или “название цвета” -** задает цвет фонастраницы. Значение задается RGB-значением или названием цвета.
* **border = число** –задает толщину рамки таблицы(в пикселах).Задание border=0 отменяет появление границы. По умолчанию рамка изображается трехмерной, но если используется параметр bordercolor тега <TABLE>, то вид рамки меняется. Когда в теге <TABLE> используется параметр border без аргументов (<table border>), то браузер отображает рамку толщиной один пиксел.
* **bordercolor = “#rrggbb” или “название цвета”** -задает цвет рамкитаблицы.
* **cellpadding = число –** задает размер интервала между границейячейки и ее содержимым. Значение по умолчанию равно 1.
* **cellspacing = число –** задает размер интервала между ячейкамитаблицы.
* **height = число, проценты** –задает высоту всей таблицы.Значениеможет быть в пикселах или в процентах от размера окна браузера.
* **width = число, проценты** –задает общую ширину таблицы.Можетбыть задан в пикселах или в процентах от размера окна браузера.

**Тег DOCTYPE**

Тег указывает стандарт разработки HTML-документа. Согласно спецификациям HTML и XHTML тег DOCTYPE (что означает «объявление типа документа») сообщает, какая именно версия (X)HTML используется на странице.

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

**Тег META**

Все мета-теги не несут визуальной информации и располагаются в секции <head></head> документа. Информация, находящаяся в этом теге, полезна для серверов, браузеров и поисковых машин. Различают два вида тегов: использующие атрибут name и использующие атрибут http-equiv. В каждом случае необходимо указывать атрибут content, который задает значение именованной информации или функции.

**Основные атрибуты тега <meta>:**

* **<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8”> -** указывает браузеру кодовую страницу для правильногоотображения символов браузером;
* **<meta http-equiv=“Content-Language” content=“ru” >–** указываетязык документ. Значение этого атрибута может использоваться как поисковыми роботами, так и веб-серверами.
* **<meta http-equiv=“Refresh” content=“время в секундах”>** или **<meta http-equiv=“Refresh” content=“время в секундах”; URL> -** в

первом случае происходит периодическая перезагрузка документа через указанное время, во втором – через указанное время

происходит переход по указанному адресу.

Автоматическую периодическую перезагрузку документа обычно назначают в случаях часто обновляемых данных (котировки акций, результаты спортивных соревнований и т.д.)

* **<meta name=“Description” content=“Описание документа, до 100**

**символов”> -** информация,содержащаяся в этом атрибуте,влияетна результаты поиска, осуществляемого поисковыми системами. Общий результат включает следующие данные:

* 1. URL документа;
  2. название документа (содержимое тега <title>);
  3. описание документа, т.е. Description или начальный фрагмент HTML-документа;
  4. рейтинг (коэффициент соответствия документа запросу пользователя)
* **<meta name=“Keywords” content=“Ключевые слова, через,**

**запятую”> -** набор слов и фраз,наиболее полно характеризующихHTML-документ, которые являются основным критерием поиска вашего документа поисковыми системами.

* **<meta name=“Author” content=“ФИО”> -** информация об автореданного документа
* **<meta name=“Copyright” content=“ ”> -** информация об авторскихправах.
* **<meta name=“Robots” content=“all”> -** правила индексации.Приуказании параметра all индексируются все HTML-страницы, при значении index – только индексные страницы.

**1.3 Таблицы**

* + своей основе таблицы представляют собой ячейки, объединенные
* строки и столбцы. Обычно таблицы используются для представления структурированных данных, однако возможности таблиц этим не ограничиваются. C помощью таблиц удобно верстать макеты веб-страниц, расположив нужным образом фрагменты контента.

Минимальный набор тегов для описания таблицы включает <TABLE>, <TR>, <TD>. Допускается вместо тега <TD> использовать тег <TH>. Текст в такой ячейке отображается браузером обычно жирным шрифтом и выравнивается по центру. В остальном, разницы между ячейками, созданными через теги <TD> и <TH>, нет.

Приводимый ниже HTML-код демонстрирует базовую структуру таблицы с четыремя ячейками:

<TABLE>

<TR>

<TD>Ячейка 1</TD>

<TD>Ячейка 2</TD>

</TR>

<TR>

<TD>Ячейка 3</TD>

<TD>Ячейка 4</TD>

</TR>

</TABLE>

Тег <TABLE> определяет содержимое таблицы. Каждая строка задается тегом <TR> и состоит из множества ячеек данных. Количество столбцов определяется количеством ячеек в строке. Ячейки с данными обозначаются тегом <TD>. Ячейка таблицы может содержать любые данные, выводимые HTML-документом (форматированный текст, гиперссылки, графические объекты, мультимедийные элементы и другие таблицы).

**Основные элементы табличной структуры**

1. **Тег-контейнер TABLE.**

**Основные атрибуты тега <table>:**

* **align = left|right|center** -задает выравнивание таблицы по краю окнабраузера.
* **background = url** -определяет изображение,которое будет служитьфоном для таблицы
* **bgcolor = “#rrggbb” или “название цвета” -** задает цвет фонатаблицы. Значение задается RGB-значением или названием цвета.
* **border = число** –задает толщину рамки таблицы(в пикселах).Задание border=0 отменяет появление границы. По умолчанию рамка изображается трехмерной, но если используется параметр bordercolor тега <TABLE>, то вид рамки меняется. Когда в теге <TABLE> используется параметр border без аргументов (<table border>), то браузер отображает рамку толщиной один пиксел.
* **bordercolor = “#rrggbb” или “название цвета”** -задает цвет рамкитаблицы.
* **cellpadding = число –** задает размер интервала между границейячейки и ее содержимым. Значение по умолчанию равно 1.
* **cellspacing = число –** задает размер интервала между ячейкамитаблицы.
* **height = число, проценты** –задает высоту всей таблицы.Значениеможет быть в пикселах или в процентах от размера окна браузера.
* **width = число, проценты** –задает общую ширину таблицы.Можетбыть задан в пикселах или в процентах от размера окна браузера.

1. **Тег-контейнер TR**

Задает в таблице строку ячеек. Строка таблицы, ограниченная тегами <tr>, не имеет никакого содержания, кроме совокупности ячеек таблицы (<td>)

**Основные атрибуты тега <tr>:**

* **align = left|right|center** -задает выравнивание текста и другихэлементов в ячейках текущей строки. В спецификации W3C признан

нерекомендуемым, предпочтение отдано позиционированию с помощью таблиц стилей (CSS).

* **valign = top|middle|bottom –** задает выравнивание текста(илидругих элементов) по вертикали в ячейках текущей строки.
* **background = url** -определяет изображение,которое будетиспользоваться как фон строки.
* **bgcolor = “#rrggbb” или “название цвета” -** задает цвет строки.

Значение цвета, заданного для фона строки, перекрывает значение, заданное для таблицы в целом.

* **bordercolor = “#rrggbb” или “название цвета”** -задает цвет рамкистроки.

1. **Тег-контейнер TD**

Определяет ячейку данных таблицы. Ячейка таблицы может содержать любую информацию, в том числе и таблицу.

**Основные атрибуты тега <td>:**

* **align = left|right|center** -задает выравнивание текста или другогосодержимого ячейки таблицы. Значение по умолчанию left.
* **valign = top|middle|bottom –** задает выравнивание текста(илидругих элементов) по вертикали в ячейках текущей строки.
* **background = url** -определяет изображение,которое будетиспользоваться качестве фона в ячейке.
* **bgcolor = “#rrggbb” или “название цвета” –** определяет цвет,

который будет использоваться в ячейке таблицы. Это значение перекрывает значения цветов, заданные на уровне строки и таблицы.

* **bordercolor = “#rrggbb” или “название цвета”** -задает цвет рамкитаблицы.
* **colspan = число –** задает количество столбцов таблицы,которыенужно объединить.
* **rowspan = число –** задает количество строк,объединяемых даннойячейкой.
* **height = число, проценты** –задает высоту ячейки.Значение можетбыть в пикселах или в процентах от высоты таблицы.
* **width = число, проценты** –задает ширину ячейки.

1. **Тег-контейнер CAPTION**

Контейнер для названия таблицы.

1. **Объединение строк и столбцов**

Сложные по структуре таблицы получаются путем объединения ячеек как по вертикале, так и по горизонтали. Для этой цели тег <td> принимает следующие атрибуты:

* + Colspan=n – объединение ячеек по горизонтали (столбцов)
  + Rowspan=n – объединение ячеек по вертикали (строк),

где n – количество ячеек.

Пример HTML-кода таблицы:

<TABLE BORDER=1 WIDTH=100% HEGHT=100% CELLPADDING=5> <TR>

<TD ROWSPAN=2>Ячейка 1</TD>

<TD COLSPAN=2>Ячейка 2</TD>

</TR>

<TR>

<TD>Ячейка 3</TD>

<TD>Ячейка 4</TD>

</TR>

</TABLE>

Результат:

|  |  |  |
| --- | --- | --- |
| Ячейка 1 | Ячейка 2 |  |
|  |  |  |
|  | Ячейка 3 | Ячейка 4 |
|  |  |  |

**1.4 Форматирование текста**

Следует помнить, что HTML разрабатывался не для строгого определения представления документа на экране, а скорее как средство разметки документа. Рассмотрим теги, которые определяют вывод элементов текста на экран.

**Абзацы и заголовки (блочные элементы)**

Блочные элементы всегда форматируются таким образом, что перед ними и после них имеется свободный интервал.

1. **Тег-контейнер P.**

Тег обозначает начало и конец абзаца.

**Основные атрибуты тега <P>:**

* **align = left|right|center** -задает выравнивание абзаца по краю окнабраузера.

1. **Теги-контейнеры H1…H6.**

Тег обозначает, что элемент является заголовком. Существует шесть различных уровней заголовков от <h1> до <h6>, каждый последующий уровень выводит заголовок меньшего размера.

**Основные атрибуты:**

* **align = left|right|center** -задает выравнивание заголовков по краюокна браузера.

1. **Тег-контейнер BLOCKQUOTE.**

Цитируемым абзацем называется вложенный текст, который выводится на экран с отступом от левого и правого полей и добавлением интервала сверху и снизу от абзаца.

**Внешний вид текста**

1. **<strong>…</strong> -** вложенный текст выводится жирным шрифтом
2. <**em> … </em>** - вложенный текст выводится курсивом
3. **<u> … </u>** -вложенный текст при выводе будет подчеркнут.
4. **<font>…</font>** -оказывает воздействие на стиль(цвет,гарнитура иразмер шрифта) вложенного текста.

**Основные атрибуты <font>:**

* **color = “#rrggbb”** –определяет цвет,используя шестнадцатеричныезначения RGB
* **face = гарнитура (или список гарнитур)** –определяет гарнутурушрифта для текста. Можно ввести список шрифтов (через запятую), браузер использует первый доступный шрифт из данного списка.
* **size = значение** –устанавливает абсолютное значение размерашрифта в диапазоне от 1 до 7 (по умолчанию 3).

**Пробелы и позиционирование**

1. **<br>** -тег конца строки.Начинает новую строку,но без пропускаинтервала.
2. **<center>…</center>** -центрирует вложенный текст горизонтально настранице.
3. **<pre>…</pre>** -определяет границы текста с заданнымформатированием. Это означает, что строки выводятся точно в том виде, как они напечатаны.

**Списки (нумерованные, маркированные)**

1. **<ol>…</ol> -** определяет начало и конец нумерованного списка,состоящего из элементов <li>.

**Основные атрибуты:**

* + **start = номер** –начало нумерации списка
  + **type= 1|A|a|I|i** –система нумерации списка

1. **<ul>…</ul> -** определяет начало и конец маркированного списка,состоящего из элементов <li>.

**Основные атрибуты:**

* + **type = disc|circle|square** –определяет тип маркера элементов списка

1. **<li>…</li>** -обозначает элемент списка.Используется внутри тегов

<ol>, <ul>.

1. **<dl>…</dl>** -задает список определений,состоящий из терминов<dt>и определений <dd>.
2. **<dt> … </dt>** -обозначает термин,включаемый в список определений.
3. **<dd>…</dd>** -обозначает определение термина в списке определений.

**1.5 Вставка изображений и других элементов**

1. <img> - вставляет изображение в документ. По умолчанию нижняя часть изображения будет выравниваться по базовой линии соседнего текста. Существует более десятка атрибутов данного тега, которые оказывают влияние на представление изображения на экране.

**Основные атрибуты:**

* + **src=url изображения** –задаетURLизображения.Обязательныйатрибут.
  + **border=n** –определяет ширину границ рамки изображения впикселах.
  + **аlt=текст** –задает строку альтернативного текста,который будетвыводиться вместо изображения, когда оно недоступно.
  + **Width=n** –определяет ширину изображения
  + **Height=n** –определяет высоту изображения
  + **Hspace=n** –задает интервал справа и слева изображения
  + **Vspace=n** –задает в пикселах размер свободного интервала над ипод выровненным графическим изображением.
  + **Clear=left|right|all** –отмена обтекания текста

1. **<embed>** -помещает на веб-страницу мультимедийные объекты,дляотображения которых на экране требуются встраиваемые приложения

(plugs-in), например Flash-movie.

Многие атрибуты аналогичны атрибутам тега <img>.

1. **<applet>…</applet>** -тег для размещения на веб-странице апплета

Java.

Многие атрибуты аналогичны атрибутам тега <img>.

1. **<bgsound>** -автоматически включает фоновый аудиофайл при загрузкедокумента в браузер.
2. **<hr>** -выводит на странице горизонтальную линейку.

**Основные атрибуты <hr>:**

**align = left|right|center –** горизонтальное выравнивание линейки **size=число -** задает толщину линейки в пикселах

**width=число или % -** задает длину линейки.

1. **<marquee>…</marquee>** -создает маркированную область«бегущего» текста.

Многие атрибуты аналогичны атрибутам тега <img>.

1. **<object>…</object>** -универсальный тег для размещения объектовсамых разных типов: элементов управления ActiveX, специальных мультимедийных встраиваемых приложений, Java-апплетов, изображений и прочих объектов.

**1.6 Создание ссылок**

<a>…</a> - тег определяет якорь внутри документа. Якорь используется для связи с другим документом. Он также может применяться в качестве фрагмента документа, который служит ссылкой для связи с определенной точкой в документе HTML.

**Основные атрибуты:**

**href = url** –ОпределяетURLдокумента,на который делается ссылка.

Примеры ссылок:

1. На локальный файл:

<a href=”filename.html”>…</a>

1. На внешний файл:

<a href=”http://server/path/file.html”>…</a>

1. На именованный якорь:

<a href=” http://server/path/file.html#fragment”>…</a>

1. На именованный якорь в том же просматриваемом документе: <a href=”#fragment”>…</a>
2. Для передачи e-mail-сообщения:

<a href=”mailto:username@domain”>…</a>

1. На файл FTP-сервера:

<a href=”ftp://server/path/filename”>…</a>

**1.8 Формы**

Формы предназначены для обмена данными между пользователем и сервером. Область применения форм не ограничена отправкой данных на сервер, с помощью скриптов можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.

Для указания браузеру, где начинается и заканчивается форма, используется тег <FORM>. Между открывающим и закрывающим тегами <FORM> и </FORM> можно помещать любые необходимые теги HTML. Это позволяет добавить элементы формы в ячейки таблицы для их форматирования, а также использовать изображения. Документ может содержать несколько форм, но они не должны быть вложены одна в другую.

HTML-теги форм лишь обеспечивают интерфейс и элементы управления для сбора информации. Реальная работа выполняется приложениями, обрабатывающими формы, обычно сценариями CGI (Common Gateway Interface, интерфейс общего шлюза). CGI представляет собой интерфейс между HTTP (программой ответственной за Web-транзакции ) и другими программами на сервере.

Программа CGI (или сценарий) может быть написана на нескольких языках программирования. Для сервера не имеет значения, на каком именно, если он обретает возможность принимать данные и посылать их назад.

Обычно Веб-дизайнер занимается разработкой элементов HTML-форм, а программирование остается опытным программистам.

**Основные элементы HTML-форм**

1. **Тег-контейнер FORM (базовая форма)**

Тег <form> обозначающий форму, содержит информацию, необходимую для взаимодействия с CGI-программой на сервере. Форма состоит из определенного числа элементов управления (полей ввода текста, кнопок, флажков, радиокнопок и т.д.) Когда пользователь завершает заполнение формы и нажимает кнопку «submit», введенные данные поступают в CGI-программу, заданную атрибутом action. Внутри одного документа может находиться несколько форм, но они не могут быть вложенными. Необходимо следить и за тем, чтобы они не перекрывали друг друга.

Каждая форма характеризуется некоторыми атрибутами которые указываются в теге <FORM>. Эти параметры задают имя формы, ее обработчик и метод отправки данных на сервер, а также некоторые другие характеристики.

**Основные атрибуты тега <form>:**

* **action** -Указывает обработчик,к которому обращаются данныеформы при их отправке на сервер. В качестве обработчика может выступать CGI-программа или HTML-документ, который включает в себя серверные сценарии. После выполнения обработчиком действий по работе с данными формы он возвращает новый HTML-документ. Если параметр action отсутствует, текущая страница перезагружается, возвращая все элементы формы к их значениям по умолчанию. В качестве обработчика можно указать также адрес электронной почты, начиная его с ключевого слова mailto. При отправке формы будет запущена почтовая программа установленная по умолчанию. В целях безопасности в браузере установлено, что отправить незаметно информацию, введенную в форме, по почте невозможно. Для корректной интерпретации данных используйте параметр enctype со значением text/plain в теге <FORM>.
* **enctype –** устанавливает тип данных отправляемых вместе с формой.Обычно устанавливать значение параметра enctype не требуется, данные вполне правильно понимаются на стороне сервера. Однако если используется поле для отправки файла (<INPUT type="file">),

следует определить параметр enctype как multipart/form-data. Допускается также устанавливать сразу несколько значений, разделяя их запятыми.

* **method -** метод сообщает серверу о цели запроса.Различают дваметода — GET и POST. Существуют и другие методы, но они пока мало используются. GET - Этот метод является одним из самых распространенных и предназначен для получения требуемой информации и передачи данных в адресной строке. Пары «имя=значение» присоединяются в этом случае к адресу после вопросительного знака и разделяются между собой амперсандом (символ &). Удобство использования метода GET заключается в том, что адрес со всеми параметрами можно использовать неоднократно, сохранив его, например, в «Избранное» браузера, а также менять значения параметров прямо в адресной строке. POST - метод POST посылает на сервер данные в запросе браузера. Это позволяет отправлять большее количество данных, чем доступно методу GET, поскольку у него установлено ограничение в 4 Кб. Большие объемы данных используются в форумах, почтовых службах, заполнении базы данных и т.д.
* **target –** после того,как обработчик формы получает данные,онвозвращает результат в виде HTML-документа. Вы можете определить окно, в которое будет загружаться итоговая веб-страница. Для этого используется параметр target, в качестве его значения используется имя окна или фрейма. Если параметр target не установлен, возвращаемый результат показывается в текущем окне. В качестве аргумента используется имя окна или фрейма, заданное параметром name. Если установлено несуществующее имя, то будет открыто новое окно. Зарезервированные имена следующие:

\_blank — загружает страницу в новое окно браузера; \_self — загружает страницу в текущее окно;

\_parent — загружает страницу во фрейм-родитель, если фреймов нет, то этот параметр работает как \_self;

\_top — отменяет все фреймы и загружает страницу в полном окне браузера, если фреймов нет, то этот параметр работает как \_self.

1. **Тег-контейнер INPUT**

Рассмотренные ниже элементы управления вводятся как варианты атрибуты type тега <input>

**Текстовое поле - <input type="text" атрибуты>** Создает элемент для ввода пользователем строки текста. **Основные атрибуты:**

* **size = “число”** –ширина поля
* **maxlength = “число”** –задает максимальное количество символовразрешенных в тексте. Если этот параметр опустить, то можно вводить строку длинее самого поля.
* **name = “Name”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.
* **value = “текст”** –задает начальный текст в поле.
* **readonly** -устанавливает режим«только для чтения».
* **disabled** -блокирование элементов форм.Обычно используется длятого, чтобы динамически с помощью скриптов изменять значение поля. Пользователь же не должен в подобном случае иметь доступ к полю, поэтому оно блокируется.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Текстовое поле ввода пароля - <input type="password" атрибуты>**

Поле для пароля — обычное текстовое поле, но отличается тем, что все символы показываются звездочками или кружочками. Хотя символы не выводятся, они доступны как данные формы, поэтому этот способ не является безопасным способом передачи паролей.

Возможные атрибуты совпадают с элементом текстового поля **Невидимый элемент ввода - <input type="hidden" атрибуты>**

Тип hidden (невидимый) добавляет в форму элемент управления, не отображаемый браузером. Он удобен, когда необходимо послать на обработку информацию вместе с данными, введенными пользователем. Это могут быть метки, используемые сценарием для сортировки форм. Некоторые сценарии CGI для обеспечения правильной работы требуют, чтобы к формам были добавлены определенные невидимые поля.

Атрибуты совпадают с атрибутами элемента текстового поля. **Флажки (галочки) - <input type="checkbox" атрибуты>**

Флажки обычно используются для задания многовариантных запросов. Когда флажок установлен, соответствующее значение передается вместе с формой в программу обработки на сервере.

**Основные атрибуты:**

* **name = “Name”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.
* **value = “текст”** –задает начальный текст в поле.
* **checked** -устанавливает элемент выбранный по умолчанию.
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Кнопка-переключатель (радиокнопка)-<input type="radio" атрибуты>**

Кнопки-переключатели (типа radio) используют, когда необходимо выбрать один единственный вариант из предложенных. Основные атрибуты:

* **name = “Name”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.
* **value = “текст”** –задает начальный текст в поле.
* **checked** -устанавливает элемент выбранный по умолчанию.
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Настраиваемая кнопка - <input type="button" атрибуты>**

Кнопка типа button не имеет какой-либо предопределенной функции, она является общим средством и настраивается JavaScript. Чтобы написать текст на кнопке, нужно задать атрибут value.

Основные атрибуты:

* **name = “Name”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.
* **value = “текст”** –задает надпись на кнопке,а также ее значение.
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Кнопка SUBMIT - <input type="submit" атрибуты>**

Когда пользователь нажимает на эту кнопку, данные формы отправляются на сервер и обрабатываются программой, указанной параметром action тега <FORM>.

Параметр name для этого типа кнопки может быть опущен. Если

значение параметра value не указывать, автоматически появится надпись

Подача запроса (для браузера Internet Explorer), Отправить (для Opera) или

Отправить запрос (для Firefox).

Основные атрибуты:

* **name = “имя”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.
* **value = “текст”** –задает начальный текст в поле.
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Кнопка RESET-<input type="reset" атрибуты>**

Кнопка reset восстанавливает данные формы в первоначальное состояние. Атрибуты совпадают с атрибутами простых кнопок. Аналогично кнопке Submit, значение параметра name можно не указывать, тогда надпись Сброс на кнопке будет добавлена браузером автоматически. Значение кнопки Reset никогда не пересылается на сервер.

**Кнопка – изображение - <input type="image" атрибуты>**

Можно изменить кнопку button произвольным изображением, используя

тип image.

**Основные атрибуты:**

* **name = “имя”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.
* **src = url – задает URL изображения. Обязательный параметр.**
* **align = left|right|center|top|bottom|middle** -задает выравниваниеполя. Результат их использования точно такой же, как и у изображений.
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Выбор файла - <input type="file" атрибуты>**

Поле формы типа file позволяет пользователю выбрать файл, хранящийся на его компьютере, и послать его на сервер как данные формы.

<form enctype="multipart/form-data" method="POST"> <input type="file" атрибуты> </form>

Параметры формы enctype="multipart/form-data" нужен для корректной передачи файла. Если его не указать, будет передан лишь путь к файлу. Атрибуты те же, что и для текстового поля.

1. **Тег-контейнер TEXTAREA** <textarea атрибуты>текст</textarea>

Тег <textarea> создает многострочное текстовое окно с прокруткой, которое позволяет пользователям вводить достаточно большой текст. Текст, который появляется между тегами <textarea>…</textarea>, будет исходным содержимым окна текстового ввода при отображении формы.

**Основные атрибуты:**

 **name = “имя”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать это поле.

* **cols = “число” –** определяет ширину(в символах).Еслипользователь вводит текст, который не помещается в созданном окне, то появляется прокуртка.
* **rows = “число”** -задает количество выводимых строк
* **wrap** –задает параметры переноса строк.Возможные значения: **off**

— отключает перенос строк; **virtuals** — показывает переносы строк, но отправляет текст как он введен; **physical** — переносы строк вставляются где указано и в таком виде текст отправляется.

* **readonly** -устанавливает режим«только для чтения».
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

1. **Тег-контейнер SELECT**

Тег <select> создает меню. Оно более компактно, чем группы кнопок или флажков. Меню выводится как спускающееся или в виде прокручиваемого списка в зависимости от заданного размера. Тег <select> работает как контейнер для любого количества тегов <option>.

**Основные атрибуты тега <select>:**

* **name = “имя”** -определяет имя поля.Предназначено для того,чтобы обработчик формы мог идентифицировать список.
* **size = “число” –** определяет количество видимых строк.Поумолчанию одна строка.
* **multiple** -задает прокручивающийся список,позволяет выбратьсразу несколько строк.
* **disabled** -блокирование элементов форм.
* **tabindex = “порядковый номер”** -определяет последовательностьперехода между полями при нажатии на <Tab>.

**Основные атрибуты тега <option>:**

* **selected** -видимое поле по умолчанию
* **value = “текст” -** определяет,что будет отправлено на сервер привыбранном пункте списка.

1. **Тег-контейнер LABEL**

Тег используется для связывания информации с элементами

управления. Каждый элемент label ассоциирован строго с одним элементом управления формы. Такая связь необходима, чтобы изменять значение переключателей и флажков при нажатии курсором мыши на текст.

Для связывания элемента формы и текста в теге <LABEL> используется параметр for, аргументом для которого служит идентификатор элемента, задаваемого параметром id.

**Основные атрибуты тега <label>:**

* **selected** -видимое поле по умолчанию
* **for = “текст” –** связывает метку с элементом управления,используязначение атрибута for, совпадающее со значением атрибута id внутри элемента управления.

**Пример:**

<form action="handler.php">

<input type="checkbox" id="t1"> Укажите на галочку <input type="checkbox" id="t2">

<label for="t2">Нажмите на этот текст, чтобы выбрать эту галочку</label>

</form>

* первой строке, чтобы активизировать флажок, надо щелкнуть на него самого. Во второй строке активным элементом является и текст, при нажатии на который курсором мыши, можно устанавливать или снимать галочку. Вот и вся разница.

1. **Теги-контейнеры FIELSET и LEGEND**

При создании сложной формы не обойтись без визуального отделения одного логического блока от другого. Этого можно добиться, используя внутри тега <FORM> сочетания тегов и стилей. Например, элементы формы можно выделить, если использовать для них фоновый цвет или рамку, задавая их через CSS. Кроме того, существует и другой подход,

который состоит в применении тега <FIELDSET>. Этот контейнер группирует элементы формы, отображая вокруг них рамку.

Чтобы добавить к рамке специальный заголовок применяется контейнер <LEGEND>, который должен располагаться в теге <FIELDSET>. Внутри тега <LEGEND> допустимо использовать текст и теги форматирования, а также стили CSS.

<form>

<fieldset>

<legend>Заголовок</legend>

<input type="checkbox"> Галочка первая<br>

<input type="checkbox"> Галочка вторая<br>

<input type="checkbox"> Галочка третья

</fieldset>

</form>

Пример HTML-кода отправки данных текстового поля на e-mail:

<FORM METHOD=POST ACTION="mailto:ling@ulstu.ru"> <INPUT TYPE=TEXT NAME="Name" SIZE=40> <INPUT TYPE=SUBMIT VALUE="Отправить"> <INPUT TYPE=RESET VALUE="Очистить"> </FORM>

1. **КАСКАДНЫЕ ТАБЛИЦЫ СТИЛЕЙ (CSS)**

CSS (Cascading Style Sheets - Каскадные таблицы стилей) – технология, позволяющая отделить параметры внешнего вида веб-документа от массива дескрипторов (тегов и атрибутов), определяющих его структуру. CSS может изменить оформление веб-страниц не затрагивая их логической структуру и информационного содержания.

Основным понятием CSS является **стиль** – т.е. набор правил оформления и форматирования, который может быть применен к различным элементам HTML-документа. Синтаксис записи CSS-правила:

Селектор {свойство1: значение; свойство2: значение; …}

**2.1 Виды CSS-селекторов**

**1. HTML-селекторы (Selectors)**

Любой элемент HTML может иметь CSS-селектор.

Самый простой селектор – просто имя элемента. После селектора располагаются фигурные скобки, содержащие набор CSS-свойств с присвоенными значениями. Селектор определяет стиль элемента, для которого он создан.

Пример: H3 {font-family:Arial; font-size:18px; color:red;}

1. **Селекторы класса (Class Selectors)**

Классы применяют, когда необходимо определить стиль для

индивидуального элемента или задать разные стили для одного тега. Для определения **селектора класса** используется обозначение **.имякласса**

При совместном использовании с тегами синтаксис для классов будет следующий:

Тег.имя класса {свойство1: значение; свойство2: значение; …}

Классы могут быть описаны без привязывания их к определенным

элементам.

Примеры:

.red {color:red;}

Любые элементы с указанным атрибутом class=“red” станут красными.

H3.red {color:red;}

Все элементы H3 с атрибутом class=“red**”** станут красными.

1. **ID-селекторы (ID Selectors)**

Идентификатор (называемый также «ID-селектор») определяет

уникальное имя элемента, которое используется для изменения его стиля

* обращения к нему через скрипты, что позволяет управлять стилем элемента динамически. Для определения ID-селектора используется обозначение **#** **имя идентификатора**.

Синтаксис использования идентификатора:

# имя идентификатора {свойство1: значение; свойство2: значение; …}

Например, был создан класс red для передачи красного цвета, но понадобился текст с курсивом. Надо определить ID, например “reditalic”.

.red {color:red;}

#reditalic {font-style: italic}

<body>

<p class=“red”>текст-красного цвета </p>

<p class=“red” id=“reditalic”> текст-курсив красного цвета </p> </body>

1. **Контекстные селекторы**

**Контекстный селектор –** селектор,работающий только в определенномконтексте. Контекстный селектор состоит из нескольких простых селекторов, разделенных пробелами.

Например, для селектора тега синтаксис будет следующий:

Тег1 Тег2 {…} – правило для Тег2 применяется только, если он является вложенным в Тег1.

1. **Группирование селекторов** Группировка:

Селектор 1, Селектор 2, … Селектор N {Описание правил стиля} Некоторые свойства имеют собственный синтаксис группирования,

связанный с заданием значений нескольких свойств в одном.

Например, для свойств font возможна запись: H3 {font: bold 14pt Arial}. Для описания стиля границ (borders), полей (margins) и отступов (padding) сокращенно описываются в таком порядке: Top, Right, Bottom, Left (верхнее, правое, нижнее, левое), такой порядок можно запомнить используя правило TRouBLEd.

1. **Наследование селекторов**

Свойства форматирования любого элемента наследуется от

элемента, в котором он размещается. CSS-свойства всегда имеют некоторые значения, даже если автор их не определил.

1. **Псевдоклассы**

Псевдоклассы – обозначение селектора для определения его

дополнительного свойства.

Синтаксис применения псевдоклассов:

Селектор:Псевдокласс {Описание правил стиля}

Вначале следует элемент, к которому добавляется псевдокласс, затем указывается двоеточие, после которого идет имя псевдокласса.

Псевдоклассы, определяющие состояние элементов – определяют текущее состояние элемента и применяют стиль к нему только в этом случае.

* **Active** –происходит при активации пользователем элемента.Например, при наведении курсора на ссылку и нажатии левой кнопки мыши.
* **Link** –применяется к непосещенным ссылкам,т.е.ссылкам,накоторые пользователь еще не нажимал.
* **Hover** –активизируется,когда курсор мыши находится в пределахэлемента, но щелчка по нему не происходит.
* **Visited** –применяется к посещенным ссылкам.Обычно такая ссылкаменяет свой цвет по умолчанию на фиолетовый, но с помощью

стилей цвет идругие параметры можно задать самостоятельно.

При этом имеет значение порядок следования псевдоклассов: задавая псевдоклассы для ссылок, делайте это всегда в таком порядке: Link, Visited, Hover, Active (этот порядок можно запомнить при помощи слов “LoVe/HAte”). Вначале указывается visited, а затем идет hover, в противном случае посещенные ссылки не будут изменять свой цвет при наведении на них курсора.

a:link {color:black;}

a:visited {color:black;}

a:hover {color:green;}

a:active {color:yellow;}

1. **Псевдостили текста**

Псевдостили применяются к некоторым элементам текстовых абзацев, например, к первой строке абзаца или букве первой строки.

* **First-line** –применяется к первой линии текста в элементе и только кблочным элементам
* **First-letter** –применяется к первой букве в элементе и только кблочным элементам

1. **Комментарии CSS**

Синтаксис комментария: /\*Это комментарий CSS\*/

**2.2 Способы задания CSS**

1. **Inline CSS –внутренние (встроенные) стили**

Определения стиля помещается в нужный тег, для этого используется специальный атрибут style. Атрибут style поддерживают практически все теги, “отвечающие” за видимые элементы страницы. Синтаксис:

style= “свойство1: значение; свойство2: значение”

Пример: <p style="font-size:12pt; font-family:Arial">Текст</p>

1. **Embedded CSS** –глобальная таблица стилей

Глобальная таблица стилей располагается в секции заголовка Веб-страницы и помещается внутрь специального тега <STYLE>: <STYLE TYPE= “TEXT/CSS”> … </STYLE>

* контейнере STYLE указываются HTML-элементы, классы и идентификаторы с их стилевыми правилами.

Пример:

<STYLE type="text/css">

body { background: url("fon.jpg")}

h3{color:33FF33; font-family:Veranda; text-align: center}

.pink{color: FF00FF}

#bold{font-weight:bold}

</STYLE>

1. **External CSS** –внешняя(связанная)таблица стилей

Стили хранятся в отдельном файле с расширением .css и привязываются к Веб-странице с помощью тега <LINK> в головной части HEAD HTML-кода. Создается CSS файл в любом текстовом редакторе (например, Блокнот), с расширением текстового файла css. Синтаксис такой же как и у глобального CSS, только в этом случае тег style не указываем.

Пример задание ссылки на css-файл:

<LINK rel="stylesheet" type="text/css" href="styles.css">

* + 1. **Import CSS**
  + теге <STYLE> можно импортировать внешнюю таблицу стилей с помощью свойства @import:

Синтаксис: @import:url(style.css)

Это свойство надо задавать в начале стилевого блока или связываемой таблицы стилей перед заданием остальных правил. Значение свойства @import является URL файла таблицы стилей. Тогда к стилям, описанным внутри <STYLE>, присоединятся стили из внешнего файла.

**Важно:** Внедренные в документCSSи ссылки на внешние файлыCSS всегда должны размещаться в самом начале документа (x)html

между тегами <HEAD></HEAD>

**Рекомендации:** Если стиль необходим в нескольких Веб-страницах,оформите его во внешней таблице стилей. Если стиль используется в нескольких местах одной Веб-страницы, переносите его в глобальную таблицу стилей. Если стиль используется в единственном месте одной Веб-страницы, внедрите его прямо в

HTML-тег.

1. **Каскадность**

Стиль определенный в глобальной таблице стилей, имеет приоритет

перед стилем из внешней таблицы. Аналогично, внутренний стиль, помещенный внутрь тега, имеет приоритет перед глобальным и внешним CSS. Это и называется каскадностью.

1. **Единицы измерения CSS**

Для задания размеров различных элементов в CSS используются

абсолютные и относительные единицы измерения. Абсолютные единицы не зависят от устройств вывода, относительные единицы определяют размер элемента относительно значения другого размера.

* Относительные единицы

Em – высота шрифта элемента

Ex – высота символа х

Px - пиксел

% - процент

* Абсолютные единицы

In – дюйм (1 дюйм равен 2,54 см) Cm - сантиметр

Mm - миллиметр

Pt – пункт (1 пункт равен 1/72)

Pc – пика (1 пика равна 12 пунктам)

**Рекомендации:** Размер шрифтов текста лучше задавать вотносительных единицах, что позволит пользователю подбирать размер, подходящий для комфортного чтения.

**2.3 Форматирование средствами CSS**

1. **Типы элементов**

* **Элементы строкового уровня (Строчные элементы)**

Элементы строкового уровня (“inline” в терминологии CSS) – это

элементы исходного документа, которые не формируют новых структурных единиц текста (например, абзацев). Их содержимое выводится в строке (например, выделенные отрывки текста в абзаце или графические объекты, импортированные в текст).

Элементы строкового уровня содержат строчные элементы. Важным свойством строчных элементов является то, что у них есть свой собственный размер. Он складывается из размера шрифта (для букв), собственных размеров картинок, расстояния между строками. Это означает, что браузер при раскладке строчных элементов определяет их размер сам, без указания параметров width и height.

Примеры строчных HTML-элементов: <a>, <img>, <em>, <span> и т.д.

* **Элементы уровня блока (Блочные элементы)**
  + элементам уровня блока (“block” в терминологии CSS) относятся элементы исходного документа, которые визуально форматируются в виде неких структурных единиц текста (например, абзацев).

Элементы уровня блока могут содержать и строчные, и блочные элементы.

Для блокового элемента можно установить значения соответствующих параметров, определяющих визуальное отображение данного элемента.

* + помощью CSS можно изменять: длину (height), ширину (width), отступы внутри (padding) и снаружи (margin), толщину границ (border); а также позиционировать блок относительно документа или другого блока.

По умолчанию ширина блока вычисляется автоматически и занимает все доступное пространство. В спецификации CSS четко указано, что ширина складывается из суммы следующих параметров: ширины самого блока (width), отступов (margin), полей (padding), и границ (border).

Если высота блока не установлена, то она вычисляется автоматически исходя из объема содержимого.

Цвет фона элемента проще всего устанавливать через универсальный параметр background. Фоном при этом заливается область, которая определяется значениями атрибутов width, height и padding. Таким образом, margin не принимает участия в формировании цветной области.

Из-за разницы в подходах браузеров при формировании блочных элементов наблюдается различие при отображении границ. Браузер Internet Explorer проводит рамку внутри блока, использует фоновую заливку по внутреннему краю границы. Различия в подходе браузеров при рисовании границ заметны только на цветном фоне и пунктирных линиях. Для сплошной рамки вид слоя в браузерах будет практически одинаковым.

Примеры блочных HTML-элементов: <h3>, <p>, <ol>, <ul>, <div> и т.д.

1. **Управление типами элементов**

Каждый элемент в модели форматирования имеет свойство display,

значение которого определяет: отображается или не отображается (none) элемент, является ли он блоком (block), списком (list-item) или встроенным элементом (inline).

1. **CSS-свойства оформления**
2. **Свойства текста**

* **font-family** -свойство гарнитуры шрифта.Задает список именгарнитур шрифтов и/или имен общедоступных гарнитур в порядке их приоритета.
* **font-style** -стиль шрифта текста.Значения: normal | italic | obliqueПо умолчанию задается свойство normal, italic задает курсив, oblique

– наклонную гарнитуру шрифта. Наклонный шрифт является разновидностью обычного шрифта. Свойство bold (жирный) в синтаксисе таблицы стилей является частью font-weight

 **font-variant** -стиль шрифта текста для задания отображения маленькими прописными буквами. Значения: normal | small-caps

* **font-weight** –задает вес,или жирность шрифта.

Значения: normal | bold | bolder | lighter | 100 | 200 |300 | … | 900

* **font-size** –свойство размера шрифта.Существует четыре способазадания размера:
  + Абсолютный размер (absolute size) – при задании размера с использованием ключевых слов размер берется относительно установленного пользователем в браузере размера medium.

Значения: xx-small | x-small | small | medium | large | x-large | xx-large

На экране компьютера умножающий фактор равен 1.5; если значение medium равно 10 pt, то large будет равно 15 pt.

* + Относительный размер (relative size) - задает размер шрифта, который на размер либо больше, либо меньше родительского шрифта соответственно.

Значения: larger | smaller

* + Длина (length) – определяет абсолютный размер шрифта Значения: em | ex | px | pt | pc | mm | cm | in
  + Проценты (percentage) – определяет абсолютный размер шрифта относительно размера шрифта родительского элемента.

38

войство font используется для определения значений свойств font-family, font-style, font-variant, font-weight, font-stretch, font-size

* **color** –свойство для задания цвета текста.
* **line-height** –свойство для задания расстояния между линиямисмежных строк текста (так называемый интерлиньяж).

Значения: normal | число | длина | процент

Значения можно задать в любых единицах измерения.

* **word-spacing** –свойство для задания интервала между словами втекстовом элементе.

Значения: normal | длина

* **letter-spacing** –свойство для задания размера интервала междусимволами

Значения: normal | длина

* **text-decoration** –свойство декорирования текста.Задаетподчеркивание, надстрочное подчеркивание, перечеркивание и мигание.

Значения: none | underline | overline | line-through | blink

* **text-transform** –свойство для задания заглавных букв в элементахтекста.

Возможные значения:

* + none – отображает элемент так, как он напечатан в HTML-источнике, и нейтрализует любые наследуемые значения.
  + сapitalize – отображает первую букву каждого слова символом верхнего регистра.
  + lowercase – отображает весь элемент символами нижнего регистра.
  + uppercase – Отображает весь элемент символами верхнего регистра.
* **text-indent** –свойство для задания размера отступа перед первойстрокой элемента текста.

Значения: длина | процент

Значения можно задать в любых единицах измерения. Также значение отступа может быть отрицательным, если первая строка должна выступать. Применяется к блочным элементам.

* **text-align** –свойство для горизонтального выравнивания текстовыхэлементов. Применяется к блочным элементам.

Возможные значения:

* + center – выравнивание по центру;
  + left – выравнивание по левому краю;
  + right – выравнивание по правому краю;
  + justify – выравнивание по ширине.
* **vertical-align** –свойство для вертикального выравнивания элемента.Возможные значения:
  + baseline – выравнивает по базовой линии родительского элемента;
  + bottom – выравнивает нижнюю часть элемента по самому низкому элементу;
  + middle – выравнивает по средней точке элемента;
  + sub – делает элемент нижним индексом;
  + super – делает элемент верхним индексом;
  + text-top – выравнивает верхнюю часть элемента по верхней части шрифта родительского элемента.
* **white-space** –определяет способ обработки пустого пространствавнутри элемента. (Устанавливает, будет ли строка, выходящая за границы элемента страницы и не содержащая пробелов, переноситься по словам).

Возможные значения:

* + normal – обычная последовательность пробелов;
  + pre – выводит все пробелы, как тег <pre> в документе HTML, но при этом не используется шрифт с фиксированной шириной;
  + nowrap – предотвращает перенос элементов текста, если не задан тег <br>.

1. **Свойства фона**

* **background-color** –свойство для задания цвета фона элемента.Применяется ко всем элементам.

Значения: RGB - значение | наименование цвета

* **background-image** –определяет фоновое изображение для элемента.Если задан и цвет фона, то изображение будет помещаться поверх цвета.

Значения: URL | none

* **background-repeat** –повторение фонового изображенияВозможные значения:
  + repeat – повторение изображения в вертикальном и горизонтальном направлениях;
  + repeat-x – повторение изображения только по горизонтали;
  + repeat-y – повторение изображения только по вертикале;
  + no-repeat – вывод изображения только один раз.
* **background-attachment** –свойство прокручивания фоновогоизображения Возможные значения:
  + scroll – перемещение фонового изображения вместе с документом;
  + fixed – фиксирование изображения относительно окна просмотра.
* **background-position** –Определяет начальное положение фоновогоизображения.

Возможные значения:

Проценты | Длина | top/center/bottom | left/center/right

* **background** –краткая форма задания всех свойств фона

1. **Свойства списка**

* **list-style-type** –задает вид маркера элемента списка.

Значения те же, что и в атрибуте type для элемента списка <li>.

* **list-style-image** –задает графическое изображение в качествемаркера элемента списка.

Значения: URL | none

* **list-style-position** –определяет положение блока маркера вструктурном блоке.

Значения: inside | outside

* **list-style** –краткая форма задания трех свойствlist-style-type, list-style-image, list-style-position в одном месте таблице стилей.

1. **БЛОЧНАЯ ВЕРСТКА <DIV>+CSS**

Основным структурным элементом блочной верстки является блочный элемент (тег-контейнер DIV), который настраивается стилями CSS. Для их корректной разметки, обязательными являются следующие свойства CSS:

* **width** и **height** -задают ширину и высотуdivэлемента;
* **float** -задаёт горизонтальную очерёдность(обтекание элементадругими слева или справа);
* **clear** -задает вертикальное обтекание(отменяет влияниегоризонтального обтекания float).

Для дальнейшего форматирования элементов используются все свойства, которые изложены в спецификации CSS 2.1.

Важная роль для позиционирования блочных элементов отведена следующим свойствам:

* **margin** -определяет размер поля(внешнее поле);
* **padding** -задает размер внутреннего отступа(отступ содержимогоблока от его рамки);
* **border** -задает размер рамки(контур элемента).

Кроме этого, часто для позиционирования блоков используют свойства top, left, right, bottom, строго задающие позицию блочного элемента, и свойство z-index, позволяющее накладывать элементы друг на друга.

**ПРАКТИЧЕСКОЕ ЗАДАНИЕ № 1.**

**Цель практического задания №1**:

1. Ознакомиться с правилами применения табличных тегов: создание простых и вложенных таблиц, объединение ячеек.
2. Изучить теги форматирования текста: заголовки, абзацы, пространственное расположение и графическое выделение текстовых элементов.

**Задание:**

Создать HTML-документ с приведенными таблицами, сохранить файл как praktika\_1.html

**Требования:**

1. Название веб-страницы «Praktika\_1»;
2. Заголовок «Практическое задание №1»;
3. Табличные заголовки (1.1, 1.2, 1.3, 1.4);
4. Рамка таблиц в 1px;
5. Содержимое ячеек имеет жирное начертание;
6. Вертикальное выравнивание элементов в ячейках;
7. В таблице 1.4 использовать вложенные таблицы, указав разную высоту и заливку.
8. Вставить текст. Создать абзацы;
9. Гарнитура шрифта текста без засечек, размер 13pt, цвет текста - #505050;

**ПРАКТИЧЕСКОЕ ЗАДАНИЕ №2**

|  |  |
| --- | --- |
| Фоновое изображение  логотип  Заголовок | |
| Ссылка меню 1 | Текст  рисунок |
| Ссылка меню 2 |
| Ссылка меню 3 |
|  |

**Цель практического задания №2**:

1. Ознакомиться с правилами табличной верстки.
2. Изучить технику вставки графических изображений: в качестве фона и отдельных изображений внутри текста.
3. Ознакомиться с основными приемами создания ссылок: относительных и абсолютных.

**Задание:**

Создать персональный сайт (не менее 3 HTML-страниц) на основе табличной верстки, сохранить файлы в папку praktika\_2.

**Требования:**

1. Ширина и высота 100%, рамка таблицы–0px, заливка ячеек;
2. Внешний отступ между ячейками – 10 px;
3. Внутренний отступ между содержимым ячейки и рамкой ячейки - 10 px;
4. Отступ между изображением и текстом – 10 px;
5. Организовать переход по страницам;
6. Создать якорную ссылку НАВЕРХ для перехода на начало страницы.
7. Гарнитура текста без засечек, выравнивание абзацев по ширине.

**ПРАКТИЧЕСКОЕ ЗАДАНИЕ №4**

**Цель практического задания №4:**

1. Отработать механизм вставки всех типов CSS в HTML-документ;
2. Поэкспериментировать с применением правил CSS.

**Задание:**

1. Применить внутренний CSS в HTML-документе практического задания №1;
2. Применить глобальный CSS в HTML-документах практического задания №2;
3. Применить внешний файл CSS в HTML-документах практического задания №3;
4. Сохранить файлы в папку praktika\_4.

**Требования:**

1. Задать правила оформления CSS тегам заголовка, тегам абзаца, тегам вставки графических изображений.
2. В глобальном и связанном CSS применить классы, идентификаторы, псевдоклассы.
3. Очистить HTML-код от оформительских атрибутов и перенести их в

CSS.

**ПРАКТИЧЕСКОЕ ЗАДАНИЕ №5**

**Цель практического задания №5:**

1. Ознакомиться с элементами форм;
2. Ознакомиться с библиотекой стандартных CGI-сценариев.

**Задание:**

Организовать любую интерактивную форму, используя элементы форм (например, анкета участника, заявка и т.д.), сохранить файл как praktika\_5.html

**Требования:**

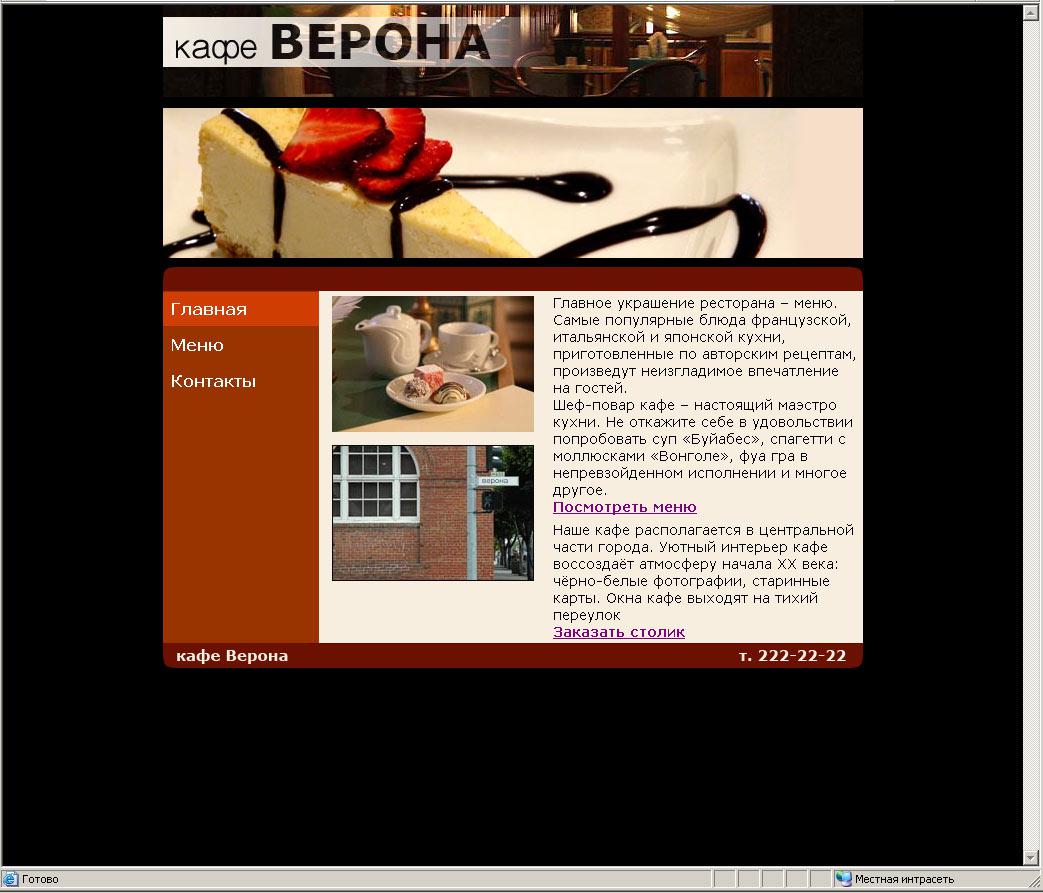
1. Обязательно использовать следующие элементы:
   * Текстовое поле ограниченной длины
   * Радиокнопки
   * Галочки
   * Выпадающий список
   * Текстовое поле для сообщения
   * Поле для пароли
2. Организовать отправку элементов форм на электронный ящик при помощи кнопки «Отправить»;
3. Создать элемент для очистки данных при помощи кнопки «Очистить»;
4. Стилевое оформление элементам форм задать при помощи глобального CSS.

**ПРАКТИЧЕСКОЕ ЗАДАНИЕ №6**

**Цель практического задания №6:**

Ознакомиться с процессом блочной верстки.

**Задание:**

Сверстать страницу, сохранить файл как index.html в папку praktika\_6

Текст для страницы находится в файле Главная.txt Все необходимые изображения в папке IMAGES

**Требования:**

1. Страница должна содержать только разметку содержания, все оформление с помощью глобального CSS.

**ПРАКТИЧЕСКОЕ ЗАДАНИЕ №7**

**Задание:**

1. Создать 2 страницы: Меню (menu.html), Контакты (contact.html).

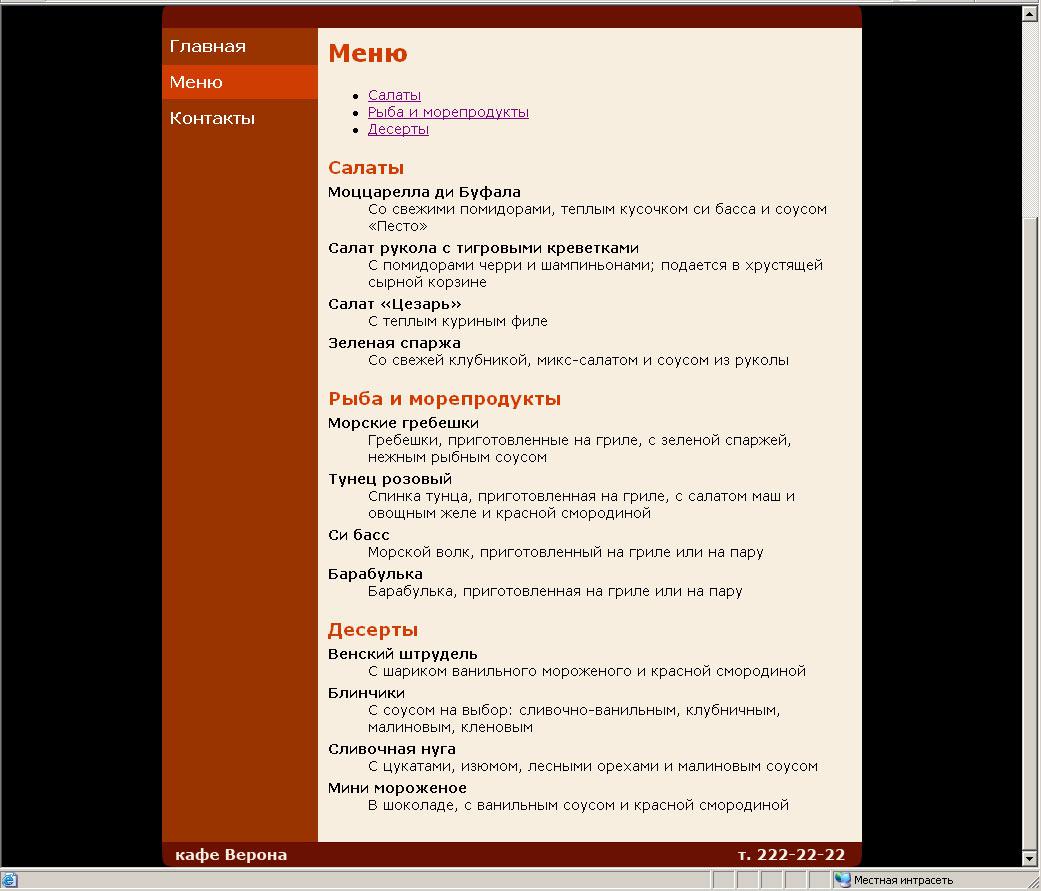
Сохранить файлы в папку praktika\_7.html

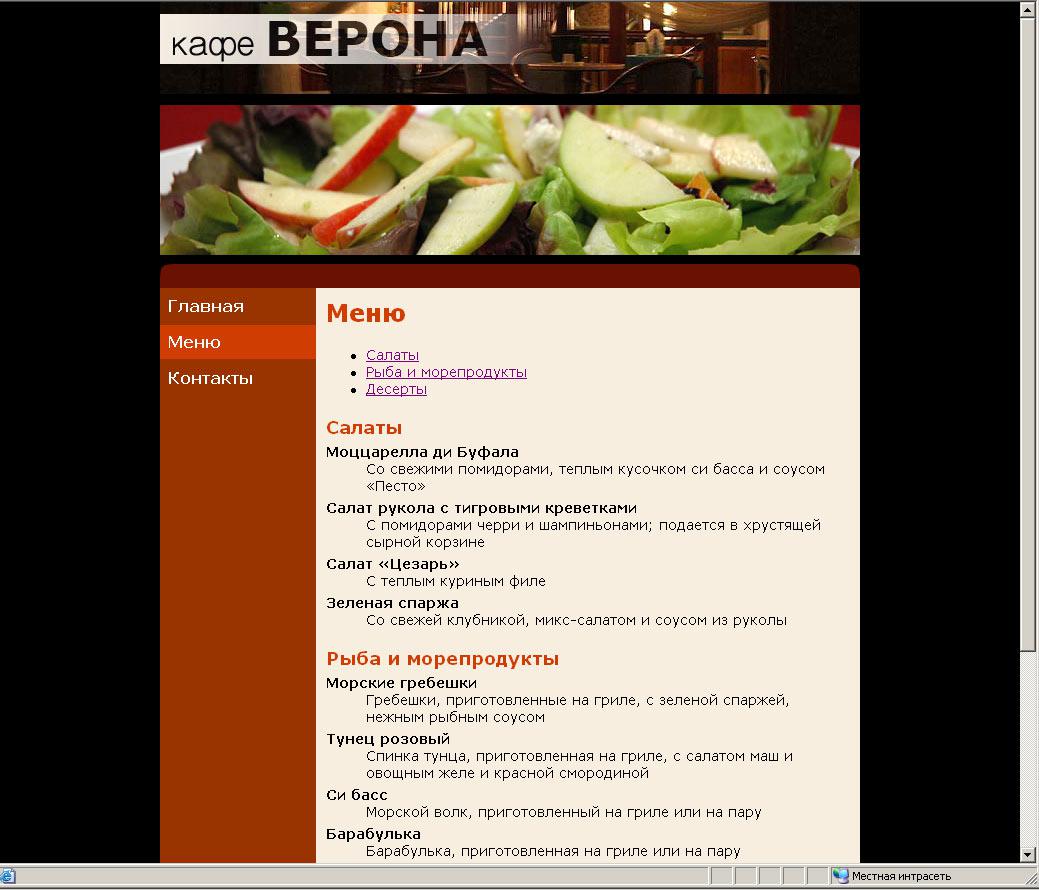
1. Скопировать index.html из практического задания №6 и вставить в папку praktika\_7
2. Преобразовать правила глобального CSS во внешний файл CSS,

сохранить как styles.css

Тексты для страниц находятся в папке texts

Все необходимые изображения в папке IMAGES





**Требования:**

1. Каждая страница обязательно должна содержать заголовок (title);
2. Каждая страница должна содержать только разметку содержания, все оформление с помощью CSS.
3. Стилевые описания, одинаковые для всех страниц, должны быть вынесены во внешний файл стилевых описаний CSS.
4. На главной странице создать гиперссылки на страницы menu.html и contact.html
5. Разделы меню оформить списком, создать локальные ссылки по разделам.
6. Применить теги заголовка 2 и 3 уровней.
7. Внутри разделов применить списки определений.

**ОСНОВНЫЕ ИСТОЧНИКИ:**

1. Алексеев А.П. Введение в Web-дизайн. - М.: СОЛОН-ПРЕСС, 2012
2. Пачевский Д.Е. Web-дизайн и Web-программирование. – ГОУ ВПО ВГТУ, 2014
3. Артанов Б. Web-мастеринг без посторонней помощи. – М.: Триумф, 2011

**ИНТЕРНЕТ-РЕСУРСЫ**

1. http://htmlbook.ru/ - Ресурс по web-мастерингу (дата обращения: 12.12.2012)
2. http://css.manual.ru/ - CSS-справочник (дата обращения: 12.12.2012)
3. http://javascript.ru/ - Центральный Javascript-ресурс. Учебник с примерами скриптов (дата обращения: 12.12.2012)
4. http://www.w3schools.com/ - Online Web Tutorial (дата обращения: 12.12.2012)