МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«25» 10 2022 г.

Web-программирование

Методические указания для выполнения лабораторных работ для студентов направления подготовки 09.03.03 Прикладная информатика

Курск 2022

УДК 004
Составитель Л.А. Лисицин

Рецензент
Кандидат технических наук, Киселев А.В.

Web-программирование: методические указания для выполнения лабораторных работ для студентов направления подготовки 09.03.03 Прикладная информатика/ Юго-Зап. гос. ун-т; сост Л.А. Лисицин, Курск, 2022. 83 с.

Содержат методические указания для выполнения лабораторных работ.

Методические указания по структуре, содержанию и стилю изложения материала соответствуют методическим и научным требования, предъявляемым к учебным и методическим пособиям.

Предназначены для студентов направления подготовки 09.03.03 Прикладная информатика.

Текст печатается в авторской редакции

Подписано в печать Формат 60 х 84 1/16.
Усл. печ. л. Уч.- изд. л. Тираж 50 экз. Заказ 2143 Бесплатно.
Юго-Западный государственный университет.
305040, Курск, ул. 50 лет Октября, 94.

Лабораторная работа № 1

Тема: Язык гипертекстовой разметки страниц HTML: общая структура документа, абзацы, цвета, ссылки; списки, графика (графические форматы, графический объект как ссылка)

Цель работы:

1. ознакомление с языком разметки гипертекстов HTML(HyperText Markup Language);
2. ознакомление с базовым синтаксисом языка, основными элементами HTML- документа;
3. приобретение навыков создания HTML- документов.

Теоретические основы

HyperText Markup Language (HTML) является стандартным языком, предназначенным для создания гипертекстовых документов в среде WEB. HTML-документы могут просматриваться различными типами WEB-броузеров. Когда документ создан с использованием HTML, WEB-броузер может интерпретировать HTML для выделения различных элементов документа и первичной их обработки. Использование HTML позволяет форматировать документы для их представления с использованием шрифтов, линий и других графических элементов на любой системе, их просматривающей.

Большинство документов имеют стандартные элементы, такие, как заголовок, параграфы или списки. Используя тэги HTML вы можете обозначать данные элементы, обеспечивая WEB-броузеры минимальной информацией для отображения данных элементов, сохраняя в целом общую структуру и информационную полноту документов. Все что необходимо, чтобы прочитать HTML-документ - это WEB-броузер, который интерпретирует тэги HTML и воспроизводит на экране документ в виде, который ему придает автор.

В большинстве случаев автор документа строго определяет внешний вид документа.

HTML-тэги могут быть условно разделены на две категории:

1. тэги, определяющие, как будет отображаться WEB-броузером тело документа в целом
2. тэги, описывающие общие свойства документа, такие как заголовок или автор документа

Основное преимущество HTML заключается в том, что документ может быть просмотрен на WEB-броузерах различных типов и на различных платформах.
HTML-документы могут быть созданы при помощи любого текстового редактора или специализированных HTML-редакторов и конвертеров.

Все тэги HTML начинаются с "<" (левой угловой скобки) и заканчиваются символом ">" (правой угловой скобки). Как правило, существует стартовый тэг и завершающий тэг. Завершающий тэг отличается от стартового прямым слешем перед текстом внутри угловых скобок. В примере тэг <TITLE> говорит WEB-броузеру об использовании формата заголовка, а тэг </TITLE> - о завершении текста заголовка.

Документ в формате HTML состоит из трех частей:

1. строки, содержащей информацию о версии HTML;
2. раздела заголовков (определяемого элементом HEAD);
3. тела, которое включает собственно содержимое документа. Тело может вводиться элементом BODYили элементомFRAMESET.

Перед каждым элементом или после каждого элемента может находиться пустое пространство (пробелы, переход на новую строку, табуляции и комментарии). Разделы 2 и 3 должны отделяться элементом HTML.

Вот пример простого документа HTML:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
"http://www.w3.org/TR/REC-html40/strict.dtd">
<HTML>
<HEAD>
<TITLE>Мой первый документ HTML</TITLE>
</HEAD>
<BODY>
<P>Всем привет!
</BODY>
</HTML>
```

Заголовочная часть документа <HEAD>

Тэг заголовочной части документа должен быть использован сразу после тэга <HTML> и более нигде в теле документа. Данный тэг представляет из себя общее описание документа. Стартовый тэг <HEAD> помещается непосредственно перед тэгом <TITLE> и другими тэгами, описывающими документ, а завершающий тэг </HEAD> размещается сразу после окончания описания документа. Например:

```
<HTML>
<HEAD>
<TITLE> Список сотрудников </TITLE>
</HEAD>
...
```

Комментарии

HTML позволяет вставлять в тело документа комментарии, которые сохраняются при передаче документа по сети, но не отображаются броузером. Синтаксис комментария:

```
<!-- Это комментарий -->
```

Комментарии могут встречаться в документе где угодно и в любом количестве.

Тэги тела документа

Тело документа <BODY>

Тело документа должно находиться между тэгами <BODY> и </BODY>. Это та часть документа, которая отображается как текстовая и графическая (смысловая) информация вашего документа.

Ссылки в HTML-документе

Для того, чтобы броузер отобразил ссылку на URL, необходимо отчертитьURLспециальными тэгами вHTML-документе. СинтаксисHTML, позволяющий это сделать - следующий:

<AHREF="URL"> текст-который-будет-подсвечен-как-ссылка </A>

Тэг <AHREF="URL">открывает описание ссылки, а тэг </A> - закрывает его. Любой текст, находящийся между данными двумя тэгами подсвечивается специальным образомWeb-броузером.

Графика внутри HTML-документа

Существует возможность включения ссылок на графические и иные типы данных в HTML-документ. Делается это при помощи тэга <IMG...ISMAP>.

Синтаксис тэга:

<IMGSRC="URL"ALT="text"HEIGHT=n1WIDTH=n2ALIGN=top|middle|bottom|texttopISMAP>

Опишем элементы синтаксиса тэга:

URL– Обязательный параметр, имеющий такой же синтаксис, как и стандартныйURL. ДанныйURLуказывает броузеру где находится рисунок. Рисунок должен храниться в графическом формате, поддерживаемом броузером.

ALT="text" – Данный необязательный элемент задает текст, который будет отображен броузером, не поддерживающим отображение графики или с отключенной подкачкой изображений.

HEIGTH=n1 – Данный необязательный параметр используется для указания высоты рисунка в пикселях. Если данный параметр не указан, то используется оригинальная высота рисунка.

WIDTH=n2 – Параметр также необязателен, как и предыдущий. Позволяет задать абсолютную ширину рисунка в пикселях.

ALIGN– Данный параметр используется, чтобы сообщить броузеру, куда поместить следующий блок текста. Это позволяет более строго задать расположение элементов на экране. Если данный параметр не используется, то большинство броузеров располагает изображение в левой части экрана, а текст справа от него.

ISMAP– Этот параметр сообщает броузеру, что данное изображение позволяет пользователю выполнять какие-либо действия, щелкая мышью на определенном месте изображения.

Пример:

<IMGSRC="http://www.softexpress.com/images/nekton.jpg"ALT="СофтСервис лого"ALIGN="top"ISMAP>

Карты сообщений:

Создание карты изображения является одной из привлекательнейших возможностей HTML, позволяющей пользователю привязывать ссылки на другие документы к отдельным частям изображений. Щелкая мышью на отдельных частях изображения, пользователь может выполнять те или иные действия, переходить по той или иной ссылке на другие документы и т.п.

Чтобы включить поддержку карты для изображения, необходимо ввести дополнительный параметр в тэг IMG:

<IMG SRC="*url*" **USEMAP="*url#map_name*"** >

Синтаксис:

<MAP NAME="map_name">- Данный тэг определяет начало описания карты с именем map_name.

<AREA...> - Описывает участок изображения и ставит ему в соответствие URL.

Параметры:

SHAPE- Необязательный параметр, указывающий на форму определяемой области изображения. Может принимать значения:

- default - по умолчанию (обычно прямоугольник)
- rect - прямоугольник
- circle - круг
- poly - многоугольник произвольной формы

COORDS- Координаты в пикселах описываемой области. Для прямоугольника это четыре координаты левого верхнего и правого нижнего углов, для круга - три координаты (две - центр круга, третья - радиус). Для многоугольника это описание каждого угла в двух координатах - соответственно число координат равно удвоенному количеству углов.

Координаты считаются с нуля, поэтому для описания области 100 на 100 используется описание:

<AREA COORDS="0,0,99,99" ...>

HREF="url"- Описание ссылки, действия по которой будут выполняться при щелчке мыши в заданной области.

NOHREF- Параметр, указывающий, что ссылка отсутствует для данного участка. По умолчанию, если не указан параметр HREF, то считается что действует параметр NOHREF. Также, для всех неописанных участков изображения считается, что используется параметр NOHREF.

Если две описанных области накладываются друг на друга, то используется ссылка, принадлежащая первой из описанных областей.

</MAP>- Завешающий тег

Пример

<MAP NAME="**map_name**">

<AREA [SHAPE=" shape "] COORDS="x,y,..." [HREF=" reference "] [NOHREF]>
</MAP>

HTMLформы

Браузеры позволяют пользователю, заполнив специальную форму, возвращающую полученное значение, выполнять некоторые действия на WWW-сервере. Когда форма интерпретируетсяWEB-броузером, создается специальные экранные элементыGUI, такие, как поля ввода,checkboxes,radiobuttons, выпадающие меню, скроллируемые списки, кнопки и т.д. Когда пользователь заполняет форму и нажимает кнопку "Подтверждение" (SUBMIT- специальный тип кнопки, который задается при описании документа), информация, введенна пользователем в форму, посылаетсяHTTP-серверу для обработки и передаче другим программам, работающим под сервером, в соответствии cCGI(CommonGatewayInterface) интерфейсом.

Когда описывается форма, каждый элемент ввода данных имеет тэг <INPUT>. Когда пользователь помещает данные в элемент формы, информация размещается в разделеVALUEданного элемента.

Синтаксис

Все формы начинаются тэгом <FORM> и звершаются тэгом </FORM>.

<FORMMETHOD="get|post"ACTION="URL">

Элементы_формы_и_другие_элементы_HTML

</FORM>

METHOD- Метод посылки сообщения с данными из формы. В зависимости от используемого метода вы можете посылать результаты ввода данных в форму двумя путями:GET: Информация из формы добавляется в конецURL, который был указан в описании заголовка формы.POST: Данный метод передает всю информацию о форме немедленно после обращения к указанномуURL.

ACTION-ACTIONописываетURL, который будет вызываться для обработки формы.

Тэги Формы

INPUT- Тэг <INPUT> используется для ввода одной строки текста или одного слова. Атрибуты тэга:

CHECKED- означает, чтоCHECKBOXилиRADIOBUTTONбудет выбран.

MAXLENGTH- определяет количество символов, которое пользователи могут ввести в поле ввода.

NAME- имя поля ввода. Данное имя используется как уникальный идентификатор поля, по которому, впоследствии, можно получить данные, помещенные пользователем в это поле.

SIZE- определяет визуальный размер поля ввода на экране в символах.

SRC-URL,. указывающий на картинку (используется совместно с атрибутомIMAGE).

TYPE- определяет тип поля ввода. По умолчанию это простое поле ввода для одной строки текста. Остальные типы должны быть явно указаны:
CHECKBOX- Используется для простых логических (BOOLEAN) значений. Значение может принимать значениеONилиOFF.
HIDDEN- Поля данного типа не отображаются броузером и не дают пользователю изменять присвоенные данному полю по умолчанию значение. Это поле используетс для передачи в программу-обработчик статической информации, как toIDпрользователя, пароля или другой информации.
PASSWORD- То же самое, что и атрибутTEXT, но вводимое пользователем значение не отображается броузером на экране.
RESET- Данный тип обозначает кнопку, при нажатии которой все поля формы примут значения, описанные для них по умолчанию.
SUBMIT- Данный тип обозначает кнопку, при нажатии которой будет вызванаCGI-программа (илиURL), описанная в заголовке формы. АтрибутVALUEможет содержать строку, которая будет высвечена на кнопке.
TEXT- Данный тип поля ввода описывает однострочное поле ввода. Используйте атрибутыMAXLENGTHиSIZEдля определения максимальной длинны вводимого значения в символах и размера отображаемого поля ввода на экране (по умолчанию принимается 20 символов).
VALUE- присваивает полю значение по умолчанию или значение, которое будет выбрано при использовании типаRADIO(для типаRADIOданный атрибут обязателен)
Меню выбора в формах

HTMLтаблицы

Таблицы в HTMLорганизуются как набор столбцов и строк. Ячейки таблицы могут содержать любыеHTML-элементы, такие, как заголовки, списки, абзацы, фигуры, графику, а также элементы форм.

Основные тэги таблицы

Таблица: <TABLE>...</TABLE>

Это основнные тэги, описывающие таблицу. Все элементы таблицы должны находиться внутри этих двух тэгов. По умолчанию таблица не имеет обрамления и разделителей. Обрамление добавляется атрибутом BORDER.

Строка таблицы: <TR>...</TR>

Количество строк таблицы определяется количеством встречающихся пар тэгов <TR>..</TR>. Строки могут иметь атрибутыALIGNиVALIGN, которые описывают визуальное положение содержимого строк в таблице.

Ячейка таблицы: <TD>...</TD>

Описывает стандартную ячейку таблицы. Ячейка таблицы может быть описана только внутри строки таблицы. Каждая ячейка должна быть пронумерована номером колонки, для которой она описывается. Если в строке отстутствует одна или несколько ячеек для некоторых колонок, то броузер отображает пустую ячейку. Расположение данных в ячейке по умолчанию

определяется атрибутами ALIGN=leftиVALIGN=middle. Данное расположение может быть исправлено как на уровне описания строки, так и на уровне описания ячейки.

Заголовок таблицы: <TH>...</TH>

Ячейка заголовка таблицы имеет ширину всей таблицы; текст в данной ячейке имеет атрибут BOLDиALIGN=center.

BORDER- Данный атрибут используется в тэгеTABLE. Если данный атрибут присутствует, граница таблицы прорисовывается для всех ячеек и для таблицы вцелом.BORDER может принимать числовое значение, определяющее ширину границы, напримерBORDER=3.

ALIGN- Если атрибутALIGNвстречается внутри <TR>, <TH> или <TD>, он управляет положением данных в ячейках по горизонтали. Может принимать значенияleft(слева),right(справа) илиcenter(по центру).

VALIGN- Данный атрибут встречается внутри тэгов <TR>, <TH> и <TD>. Он определяет вертикальное размещение данных в ячейках. Может принимать значенияtop(вверху),bottom(внизу),middle(по середине) иbaseline(все ячейки строки прижаты кверху).

NOWRAP- Данный атрибут говорит о том, что данные в ячейке не могут логически разбиваться на несколько строк и должны быть представлены одной строкой.

COLSPAN- Указывает, какое количество ячеек будет объединено по горизонтали для указанной ячейки. По умолчании - 1.

ROWSPAN- Указывает, какое количество ячеек будет объединено по вертикали для указанной ячейки. По умолчании - 1.

COLSPEC- Данный параметр позволяет задавать фиксированную ширину колонок либо в символах, либо в процентах, напримерCOLSPEC="20%".

Пример таблицы

```
<TABLE BORDER=5>
<CAPTIONALIGN=bottom> Таблица №1 </CAPTION>
<TR><TDROWSPAN=2></TD><TH COLSPAN=2>Среднее
значение</TH></TR>
<TR><TH>Рост</TH><TH>Вес</TH></TR>
<TR><TD>Мужчины</TD><TD                    ALIGN=center>174</TD><TD
ALIGN=center>78</TD></TR>
<TR><TD>Женщины</TD><TD                    ALIGN=center>165</TD><TD
ALIGN=center>56</TD></TR>
</TABLE>
```

HTML фреймы

Фреймы, позволяющие разбивать Web-страницы на множественные скроллируемые подокна, могут значительно улучшить внешний вид и функциональность Web-приложений. Каждое подокно, или фрейм, может иметь следующие свойства:

1. Каждый фрейм имеет свой URL, что позволяет загружать его независимо от других фреймов
2. Каждый фрейм имеет собственное имя (параметр NAME), позволяющее переходить к нему из другого фрейма
3. Размер фрейма может быть изменен пользователем прямо на экране при помощи мыши (если это не запрещено указанием специального параметра)

Формат документа, использующего фреймы, внешне очень напоминает формат обычного документа, только вместо тэга BODY используется контейнер FRAMESET, содержащий описание внутренних HTML-документов, содержащий собственно информацию, размещаемую во фреймах.

<FRAMESET COLS="value" | ROWS="value">
<FRAME SRC="url1">
<FRAME ...> ...
</FRAMESET>

FRAMESET

<FRAMESET [COLS="value" | ROWS="value"]>

Тэг <FRAMESET> имеет завершающий тэг </FRAMESET>. Все, что может находиться между этими двумя тэгами, это тэг <FRAME>, вложенные тэги <FRAMESET> и </FRAMESET>, а также контейнер из тэгов <NOFRAME> и </NOFRAME>, который позволяет строить двойные документы для броузеров, поддерживающих фрэймы и не поддерживающих фрэймы.

Данный тэг имеет два взаимоисключающих параметра: ROWS и COLS.

ROWS="список-определений-горизонтальных-подокон"- Данный тэг содержит описания некоторого количества подокон, разделенные запятыми. Каждое описание представляет собой числовое значение размера подокна в пикселах, процентах от всего размера окна или связанное масштабное значение. Отсутствие атрибута ROWS определяет один фрэйм, величиной во все окно броузера.

Синтаксис используемых видов описания величин подокон:

value- Простое числовое значение определяет фиксированную высоту подокна в пикселах.

value%- Значение величины подокна в процентах от 1 до 100. Если общая сумма процентов описываемых подокон превышает 100, то размеры всех фрэймов пропорционально уменьшаются до суммы 100%. Если, соответственно, сумма меньше 100, то размеры пропорционально учеличиваются.

value*- Вообще говоря, значение value в данном описании является необязательным. Символ "*" указывает на то, что все оставшееся место будет принадлежать данному фрэйму.

COLS="список-определений-горизонтальных-подокон"- То же самое, что и ROWS, но делит окно по вертикали, а не по горизонтали.

Внимание! Совместное использование данных параметров может привести к непредствазуемым результатам. Например, строка: <FRAMESET ROWS="50%,50%" COLS "50%,50%"> может привести к ошибочной ситуации.
Пример:
<FRAMESET COLS="50,*,50">- описывает три фрэйма, два по 50 точек справа и слева, и один внутри этих полосок.

FRAME

<FRAME SRC="*url*" [NAME="frame_name"] [MARGINWIDTH="nw"] [MARGINHEIGHT="nh"] [SCROLLING=yes|no|auto] [NORESIZE]>
Данный тэг определяет фрэйм внутри контейнера FRAMESET.

SRC="url"

Описывает URL документа, который будет отображен внутри ланного фрэйма. Если он отсутствует, то будет отображен пустой фрэйм.

NAME="frame_name"

Данный параметр описывает имя фрэйма. Имя фрэйма может быть использовано для определения действия с данным фрэймом из другого HTML-документа или фрэйма (как правило, из соседнего фрэйма этого же документа). Имя обязательно должно начинаться с символа. Содержимое поименованных фрэймов может быть задействовано из других документов при помощи специального атрибута TARGET, описываемого ниже.

MARGINWIDTH="value"

Это атрибут может быть использован, если автор документа хочет указать величину разделительных полос между фрэймами сбоку. Значение *value*указывается в пикселах и не может быть меньше единицы.

MARGINHEIGHT="value"

То же самое, что и MARGINWIDTH, но для верхних и нижних величин разделительных полос.

SCROLLING="yes | no | auto"

Этот атрибут позволяет задавать наличие полос прокрутки у фрэйма.

NORESIZE

Данный атрибут позволяет создавать фрэймы без возможности изменения размеров.

NOFRAMES

Данный тэг используется в случае, если создается документ, который может просматриваться как броузерами, поддерживающими фрэймы, так и броузерами, их не поддерживающими. Данный тэг помещяется внутри контейнера FRAMESET, а все, что находится внутри тэгов <NOFRAMES> и </NOFRAMES> игнорируется броузерами, поддерживающими фрэймы.
Примеры

```
<FRAMESET ROWS="*,*">
<NOFRAMES>
<H1>Ваша версия WEB-броузера не поддерживает фрэймы!</H1>
```

```
</NOFRAMES>
<FRAMESET COLS="65%,35%">
<FRAME SRC="link1.php">
<FRAME SRC="link2.php">
</FRAMESET>
<FRAMESET COLS="*,40%,*">
<FRAME SRC="link3.php">
<FRAME SRC="link4.php">
<FRAME SRC="link5.php">
</FRAMESET>
</FRAMESET>
```

Задание:

Создайте документ, состоящий из 2 горизонтальных фрэймов. В правом фрэйме разместите изображение, например компьютера. По нажатию на отдельных элементах компьютера во втором фрэйме появляется описание данного элемента.

Контрольные вопросы

1. На какие части разделяется HTML-документ?
2. При помощи какого тэга в HTML-документ добавляется графика?
3. Назовите основные тэги формы?
4. Для чего используется карта сообщений?
5. Для чего используется фрэйм NOFRAMES?

Лабораторная работа №2

Тема: Создание простейшего HTML-документа. Форматирование шрифта и абзаца

Цель работы:

1. ознакомление с созданием простейшего HTML- документа;
2. ознакомление с форматирование шрифта и абзаца HTML- документа;
3. приобретение навыков создания HTML- документа.

Задание на выполнение

1. Создать файл с гипертекстовым документом:

● Запустить редактор Блокнот, ввести текст:

Приветствую Вас на моей первой web-страничке!

● Сохранить файл в созданной папке. При сохранении, в окне диалога **Сохранить как…** в строке **Тип файла:** выбрать вариант **Все файлы (*.*)** , а в строке **Имя файла** задать имя с расширением .**htm**, например **1_name.htm** (где **name** – ваше имя)

● Закрыть документ, найти его пиктограмму в окне **Мой компьютер** или в окне програм-
мы **Проводник.**

● Открыть файл. Проанализировать, *с помощью какого приложения отображается файл* и как выглядит введенная фраза.

2. Ввести теги, определяющие структуру html-документа:

● С помощью контекстного меню открыть файл с помощью редактора Блокнот. Ввести приведенные ниже теги, в разделе заголовка документа (между тегами <**TITLE**> </**TITLE**>) указать свою фамилию.

<HTML>
<HEAD> <TITLE> Фамилия </TITLE>
</HEAD>
<BODY>
Приветствую Вас на моей первой web-страничке!
</BODY>
</HTML>

● **Сохранить** документ под тем же именем, обновить его отображение в броузере (выполнить **Вид/Обновить** или нажать кнопку **Обновить** на панели

инструментов). Проанализировать произошедшие изменения в отображении документа.

3.Отредактировать документ:

● Вызвать меню броузера **Вид/Просмотр HTML-кода** и добавить после текста **«Приветствую Вас на моей первой web-страничке!»** текст подписи:

Студент группы NNN Фамилия Имя

Сохранить документ (но не закрывать) и обновить его просмотр в броузере.

● Используя одиночный тег **
, отредактировать документ так, чтобы подпись начиналась с новой строки, а **Фамилия Имя – в следующей строке. Просмотреть в броузере новый вариант.

Внимание! ***После каждого изменения документ нужно сохранять, а просмотр в броузере начинать с обновления загрузки документа с помощью кнопки «Обновить» на панели инструментов.***

4. Оформить фрагменты текста с помощью стилей **Заголовков**:

●Первую строку документа оформить стилем **Заголовок 1-го уровня** с помощью парного тега **<H1> …</H1>**. Вторую строку оформить как **Заголовок 6-го уровня**, а третью как **Заголовок 4-го уровня**.

● Просмотреть документ в броузере, изменяя настройку отображения шрифтов (меню **Вид / Размер шрифта / Самый крупный, Средний, Мелкий** и **Самый мелкий**).

● Поменять стиль оформления первой строки на **Заголовок 2 уровня**, второй строки - на **Заголовок 5 уровня**, последней строки - на **Заголовок 3-го уровня**.

5. Выполнить форматирование шрифта:

● После строки **Фамилия Имя** добавить еще одну строку текста

Нас утро встречает прохладой

● Оформить фразу по приведенному ниже образцу.

В слове УТРО все буквы должны иметь **разные цвета**. В слове ПРОХЛАДОЙ оформить буквы ПРО – **красным** цветом, ОЙ – **синим**.

● Оформить строку с подписью (**Студент группы NNN Фамилия Имя**) **курсивом**, размер шрифта задать относительным изменением. Использовать теги **<FONT SIZE=«+2»>** и **<I>**

● Просмотреть полученный документ в броузере.

5. Выполнить форматирование абзацев:

● Создать новый документ **2_name.htm**, сохранить его в той же рабочей папке.

● Ввести текст (использовать копирование текста из документа **1_name.htm**):

<HTML>
<HEAD> <TITLE> Фамилия </TITLE>
</HEAD>
<BODY>
**Приветствую Вас на моей второй web-страничке!
 Монолог Гамлета**
</BODY>
</HTML>

● Выровнять текст **по центру**.

●Ввести текст:

Быть иль не быть - вот в чем вопрос. Что благороднее: сносить удары неистовой судьбы - иль против моря невзгод вооружиться, в бой вступить. И все покончить разом...

● Оформить выравнивание абзаца **по ширине**.

● Ограничить абзац горизонтальными разделительными линиями сверху и снизу, используя тег <HR>.

● Скопировать монолог и разбить его на абзацы. Выровнять **по центру**.

Быть иль не быть - вот в чем вопрос.
Что благороднее: сносить удары
Неистовой судьбы - иль против моря
Невзгод вооружиться, в бой вступить
И все покончить разом...

● Сохранить документ.

● Просмотреть документ в окне броузера, изменяя размер окна.

6. Выполнить оформление списков:

● Создать новый документ **3_name.htm**, сохранить его в той же рабочей папке жесткого диска.

● Ввести текст:

<HTML>
<HEAD> <TITLE> Фамилия </TITLE>
</HEAD>
<BODY>
Приветствую Вас на моей третьей web-страничке!
</BODY>
</HTML>

● Дополнить текст документа (между тегами <BODY>…</BODY>) следующим текстом:

Я знаю как оформлять:
Шрифты,
Заголовки,
Абзацы

● Оформить три последние строки как **список нумерованный**. Для этого использовать следующую конструкцию тегов:

<OL>
<LI> Шрифты, </LI
<LI> Заголовки, </LI>
<LI> Абзацы </LI>
</OL>

● Поменять оформление списка на **список маркированный**. Использовать теги <UL>, </UL>

●Создать «смешанный» список:

Я **знаю как оформлять:**

1. Шрифты

● Размер
● Цвет
● Гарнитуру
● Индексы

2. Заголовки

● От 1-го до 6-го уровня

3. Абзацы

● Выравнивание

● Разрыв строк внутри абзаца

● С использованием переформатирования.

7. Предъявить результаты работы преподавателю.

Таблица основных тегов HTML-документа. Теги форматирования шрифта и абзаца

Назначение	Вид тегов	Примечание
Общая структура документа HTML		
Тип документа	<HTML></HTML>	Начало и конец документа
Имя документа	<HEAD></HEAD>	Не отображается броузером
Заголовок	<TITLE></TITLE>	Содержимое строки заголовка ок- на броузера
Тело документа	<BODY></BODY>	Содержимое WEB-страницы

Структура содержания документа		
Внутренние заго- ловки различного уровня	<H№> текст </H№>	Где № – номер уровня заголовка (от 1 до 6). Например, <H1>…</H1> - заголовок 1-го уровня.
Заголовок с выравниванием	<H№ ALIGN=”LEFT| CENTER| RIGHT”> текст </H№>	LEFT - по левому краю, CENTER - по центру, RIGHT - по правому краю.
Форматирование абзацев		
Создание абзаца (параграфа)	<P> текст </P>	Абзацы отделяются двойным межстрочным интервалом
Перевод строки внутри абзаца	 	Одиночный тег
Выравнивание аб- заца	<P ALIGN=”LEFT”>текст </P> <P ALIGN= ”CENTER”>текст </P> <P ALIGN=”RIGHT”> текст</P> <P ALIGN=”JUSTIFY”> текст </P>	LEFT - по левому краю CENTER - по центру RIGHT - по правому краю JUSTIFY – по ширине
Разделительная горизонтальная линия между аб- зацами	<HR SIZE=«?»>	Одиночный тег. «?» - толщина линии в пикселях. Толщину ли- нии можно не указывать.
Форматирование шрифта		
Жирный	<B> текст </B>	<B>**Жирный**</B>
Курсив	<I> текст </I>	<I> *Курсив* </I>

Подчеркнутый	<U> текст </U>	<U> Подчеркнутый </U> <S Перечеркнут > ый</S>
Перечеркнутый	<S> текст </S>	
Увеличенный размер	<BIG> текст </BIG >	
Уменьшенный размер	<SMALL> текст </SMALL>	
Верхний индекс	^{текст}	^{Верхний индекс} _{Нижний индекс }
Нижний индекс	_{текст}	
Размер шрифта	<FONT SIZE=«?»> текст </FONT>	?- значения от 1 до 7 или относи- тельное изменение (например, +2)
Базовый размер шрифта	<BASEFONT SIZE=«?»>	Одиночный тег ? – размер от 1 до 7; по умолча- нию равен 3 и задается для всего документа в целом
Гарнитура шрифта	<FONT FASE=«название1, название2, …»> текст </FONT>	Текст оформляется первым, уста- новленным на компьютере шриф- том из списка названий
Цвет шрифта	<FONT COLOR=”$$$$$$”> текст </FONT>	Цвет задается либо ключевым словом, либо шестнадцатеричным кодом с символом # RED –красный, #FF0000 – шест- надцатеричный код – красного цвета
Создание списков		

Нумерованный	<OL>элементы списка</OL>	<OL> <LI>Элемент списка 1 </LI> <LI>Элемент списка 2</LI> <LI>Элемент списка 3</LI> </OL>
Маркированный	<UL> элементы списка </UL>	
Элемент списка	<LI> элементы списка </LI>	

Таблица основных цветов

Цвет	**Color's name**	**Шестнадцатеричный код цвета**		
		Red	**Green**	**Blue**
Черный	black	00	00	00
Темно-синий	navy	00	00	80
Голубой	blue	00	00	FF
Зеленый	green	00	80	00

Цвет	**Color's name**	**Шестнадцатеричный код цвета**		
		Red	**Green**	**Blue**
Темно-зеленый	teal	00	80	80
Салатный	lime	00	FF	00
Бледно-голубой	aqua	00	FF	FF
Вишневый	maroon	80	00	00
Фиолетовый	purple	80	00	80
Оливковый	olive	80	80	00
Серый	gray	80	80	80

Светло-серый	silver	C0	C0	C0
Красный	red	FF	00	00
Лиловый	fushsia	FF	00	FF
Желтый	yellow	FF	FF	00
Белый	white	FF	FF	FF

Вставка в HTML-документ рисунков. Создание закладок и гиперссылок

Задание на выполнение

1. Скопировать из Интернета или какой-либо папки в личную папку файлы три графических файла (например, **Arrows1.wmf, Arrows2.wmf, Arrows3.wmf**).
   *Убедиться, что созданные ранее документы **1_name.htm, 2_name.htm и 3_name.htm также** находятся в вашей папке на жёстком диске.*
2. Вставка рисунков в документ.
   - Открыть в Блокноте документ **2_name.htm.**
   - Вставить рисунок **Arrows1.wmf** в начало документа **2_name.htm**. Для вставки использовать тег **IMG** с параметрами **WIDTH** и **HEIGHT** для установки размеров рисунка 50 пикселов по горизонтали и по вертикали.
   - Сохранить документ под именем **4_name.htm**.
   - Просмотреть в броузере полученный результат.
   - Ввести в тег рисунка параметр **ALIGN** для выравнивания рисунка по правому краю. Просмотреть результат в броузере.
   - Вставить рисунок **Arrows2.wmf** в конец документа **4_name.htm** перед, подобрать тип выравнивания рисунка на свое усмотрение. Установить размер рисунка 100 пикселов по горизонтали и по вертикали. С помощью параметра **ALT** создать всплывающую подсказку **«Рисунок 2»**, появляющуюся при наведении курсора мыши на рисунок.
   - Просмотреть в броузере полученный результат.
3. Создание гиперссылок и закладок.
   - В документе **3_name.htm** закрепить гиперссылки за следующими элементами списка:
     За словом **Шрифт** – гиперссылка на документ **1_name.htm.**
     За словом **Заголовки** – на документ **1_name.htm.**
     За словом **Абзацы** - на документ **2_name.htm.**
   - Создать закладку в документе **1_name.htm** перед фразой **«Нас утро встречает прохладой»**. Дать ей имя «**Morning»**.
   - Изменить первую гиперссылку (слово **Шрифт**) так, чтобы она указывала на закладку

«**Morning»** в документе **1_name.htm.**

● Создать закладку в начале текущего документа **3_name.htm**.. Присвоить ей имя «**Hello»**.

● Изменить вторую гиперссылку (на слове **Заголовки**), определив для неё переход в начало текущего документа на установленную закладку «**Hello»**.

●Создать закладку в документе

2_name.htm перед фрагментом монолога. Присвоить ей имя «**Mono»**.

● Установить на слово **переформатирования** гиперссылку на закладку «**Mono»**.

● Проверить правильность переходов по всем гиперссылкам.

3. Закрепить гиперссылки за графическими файлами:

● Отредактировать тег вставки рисунка **Arrows1.wmf,** ввести в тег атрибут **ALT** для отоб-ражения текста подсказки **«Вернуться»**. Просмотреть в броузере как реагирует рисунок на наведение курсора мыши.

● Закрепить за рисунком **Arrows1.wmf** в документе **4_name.htm** гиперссылку на документ **3_name.htm**. Выполнить переход между документами.

4. Предъявить результат преподавателю.

Основные теги вставки рисунков, закладок и гиперссылок

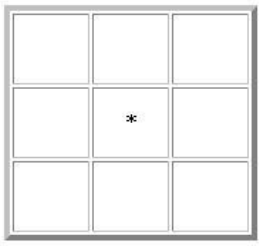
Вставка изображений		
Вставка графиче-ского файла	<IMG SRC=”файл” WIDTH=”ширина” HEIGHT=”высота” >	*Пример*: <IMG SRC=”grafica.gif” WIDTH=”550” HEIGHT=”240” ALIGN=” right” ALT=”Графический файл”>
Выравнивание картинки относи-тельно текста	<IMG SRC=”файл” ALIGN=”left|right|top|bottom”>	
Вывод текста всплывающей подсказки при наведении курсора мыши на	<IMG SRC=”файл” ALT=”текст”>	

рисунок		
Вставка ссылок		
Ссылки на другую страницу	<A HREF=” страница”> текст </A>	<A HREF=”str2.htm”> Ссылка1</A>
Ссылка на закладку в другом документе	<A HREF=” страница # имя закладки”> текст</A>	<A HREF=” index.htm #met1”> На главную страницу </A>”
Ссылка на закладку в том же документе	<A HREF=”#имя закладки ”> текст </A>	<A HREF=” #metka1”> Ссылка2</A>
Определить заклад-ку	<A NAME=”имя закладки”>текст</A>	<A NAME=”metka1”></A>
Цвет фона, текста и ссылок		
Фоновая картинка	<BODY BACKGROUND=”файл рисун-ка”>	<BODY BACKGROUND =”grafica.gif”
Цвет фона	<BODY BGCOLOR=”#$$$$$$”>	TEXT=”black”(черный)
Цвет текста	<BODY TEXT=”#$$$$$$”>	LINK=”#FF0000” (красный)
Цвет ссылки	<BODY LINK=”#$$$$$$”>	VLINK=”#FFFF0 0” (желтый)
Цвет пройденной ссылки	<BODY VLINK=”#$$$$$$”>	ALINK=”#FFFFF F” (белый)
Цвет активной ссылки	<BODY ALINK=”#$$$$$$”>	</BODY>

Создание и форматирование таблиц

Задание на выполнение

1. Создать таблицу по приведенному образцу, сохранить документ под именем **tabl_name.htm.** Сверху над таблицей разместить заголовок **Таблица №1**



При отображении таблицы в броузере должны удовлетворяться следующие условия:

●таблица должна быть выравнена по центру и быть правильной (симметричной) формы;

● в центральной ячейке поместить символ ***** (звездочка), остальные ячейки должны быть пустыми.

Примечание. Для отображения пустых ячеек в них нужно поместить символьный примитив пробела ** **;

2. В этом же документе создать копию таблицы №1, ввести заголовок **Таблица №2** и модифицировать ее:

● В центральной ячейке разместить рисунок **Arrows3.wmf**

● «Раскрасить» все остальные ячейки в различные цвета.

3. Создать еще одну копию таблицы – **Таблица №3** и отредактировать теги таблицы так, что-бы она соответствовала приведенному ниже образцу.

Примечание. Для объединения ячеек в тегах <TD> необходимо использовать параметры **colspan=** и **rowspan=**

4. Создать новый HTML-документ - **rasp_name.htm** с расписанием занятий.

● Документ должен начинаться заголовком **Расписание занятий гр. NNN на весенний семестр 2005 г.**

● Первая строка таблицы должна быть оформлена как заголовки полей (с использованием тегов **<TH>**).

● Таблица по ширине должна занимать полный размер окна. Ширину отдельных столбцов задать в относительных единицах (в %), с тем, чтобы при изменении ширины окна про-порции таблицы сохранялись.

День недели	Время	Предмет	Преподаватель	Аудитория
Понедельник	8:30-10:05	Математика (лек)	доц. Иванов А.А.	320
	10:15-11:50	Математика (пр)	преп. Петрова И.А.	302
	12:30-14:05	Физика (лаб)	доц. Сидоров О.И.	307
Вторник	8:30-10:05	История (лек)	проф. Громова О.А.	310
	10:15-11:50	История (сем)	преп. Попов М.А.	302
	12:30-14:05	Физика (лаб)	доц. Сидоров О.И.	307
...	...	...	...	...

● Просмотреть созданный документ в броузере при различных размерах окна и различных настройках размера шрифта.

5. Сохранить файл с расписанием под именем **rasp_menu_name.htm** и модифицировать его.
6. После заголовка создать таблицу, состоящую их одной строки меню с названиями дней недели.

Расписание

Понед.	Вторник	Среда	Четверг	Пятн.	Субб.

7. В таблице с расписанием установить закладки на названия дней недели.
8. В таблице меню создать гиперссылки на соответствующие дни недели.
9. Выполнить цветовое оформление каждой ячейки меню.
10. Проверить правильность выполнения переходов по гиперссылкам.

11. Создать группу web-страниц, объединенных меню:

● На рабочем диске создать папку **My_raspisanie** для размещения файлов расписания.

● Поместить расписание на каждый день недели и таблицу с меню в отдельные файлы. Имена файлов: **menu.htm** – для главной страницы, названия дней недели – для остальных. Все документы разместить в папке **My_raspisanie**.

● Отредактировать гиперссылки меню так, чтобы по ним выполнялись переходы на соответствующий документ.

● В конце каждого файла с расписанием на день организовать гиперссылку для возврата в главный документ с меню.

● Оформить фон каждого дня недели собственным цветом, совпадающим с цветом ячейки таблицы меню.

12. Предъявить результат преподавателю.

Теги оформления таблиц

Определить таблицу	<TABLE></TABLE>	Пример <TABLE border=”1” align =«CENTER» width=«50%» > <TR> <TH >Товар</TH> <TH>Цена</TH > </TR> <TR> <TD>Радиотелефон</ TD> <TD>2000 </TD> </TR> </TABLE>
Окантовка таблицы	<TABLE BORDER =”?” </TABLE>	
Строка таблицы	<TR> </TR>	
Выравнивание	<TR ALIGN=left | right | center | middle | bottom >	
Ячейка таблицы	<TD></TD>	
Выравнивание по горизонтали	<TD ALIGN=LEFT | RIGHT | CEN-TER>	
Выравнивание по вертикали	<TD VALIGN = TOP|MIDDLE| BOT-TOM>	
Установка ширины ячейки (в пикселях или %)	<TD WIDTH=«?»>	

Товар	**Цена**

		Радиотелефон 2000
Заливка цветом ячейки	<TD BGCOLOR = «# цвет»> </TD>	<TD BGCOLOR = «#FF0000»> </TD> красный цвет
Заголовок столбца или строки	<TH>текст </TH>	Текст в ячейке выравнивается по центру, устанавливается жирный шрифт

Контрольные вопросы

1. С помощью какого приложения отображается файл и как выглядит введенная фраза?
2. Перечислите основные теги вставки рисунков.
3. Перечислите основные теги вставки закладок.
4. Перечислите основные теги вставки гиперссылок.

Лабораторная работа №3

Тема: Использование стиля при оформлении сайта. Спецификации CSS1, CSS2

Цель работы:

а) ознакомление с каскадными таблицами стилей (CSS);

б) ознакомление с базовым синтаксисом, основными элементами CSS - документа;

в) приобретение навыков создания HTML – документов с использованием CSS.

Теоретические основы

Расшифровывается CSS (англ. *Cascading Style Sheets)* как каскадные таблицы стилей и является технологией оформления веб-страниц.

Основным понятием CSS является стиль – т. е. набор правил оформления и форматирования, который может быть применен к различным элементам документа. В стандартном HTML для присвоения какому-либо элементу определенных свойств (таких, как цвет, размер, положение на странице и т. п.) приходилось каждый раз описывать эти свойства, увеличивая размер файла и время загрузки на компьютер просматривающего ее пользователя.

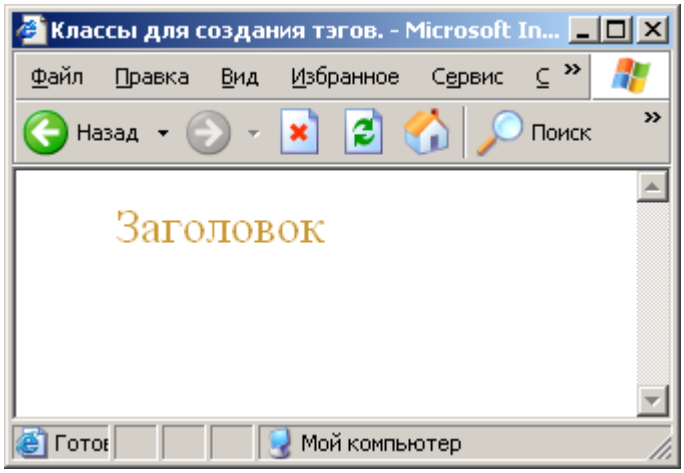
CSS действует более удобным и экономичным способом. Для присвоения какому-либо элементу определенных характеристик необходимо один раз описать этот элемент и определить это описание как стиль, а в дальнейшем просто указывать, что элемент, который нужно оформить соответствующим образом, должен принять свойства указанного стиля.

Более того, можно сохранить описание стиля не в тексте кода документа, а в отдельном файле – это позволит использовать описание стиля на любом количестве Web-страниц, а также изменить оформление любого количества страниц, исправив лишь описание стиля в одном (отдельном) файле.

Кроме того, CSS позволяет работать со шрифтовым оформлением страниц на гораздо более высоком уровне, чем стандартный HTML, избегая излишнего утяжеления страниц графикой.

```
<html>
<head>
<style type="text/css">
        .newfont{font-size:24px; color:#CC9933}
</style>
<title>Классы для создания тэгов.</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
```

```
</head>
<body>
<blockquote class=" newfont ">Заголовок</blockquote>
</body>
</html>
```



Данный пример иллюстрирует вариант объявления нового стиля в документе и потом его использования.

Синтаксис и элементы CSS

1 Добавление стилей CSS в HTML-документ

Существует несколько способов связывания документа и таблицы стилей:

– Связывание - позволяет использовать одну таблицу стилей для форматирования многих страниц HTML

– Внедрение - позволяет задавать все правила таблицы стилей непосредственно в самом документе

– Встраивание в теги документа - позволяет изменять форматирование конкретных элементов страницы

– Импортирование - позволяет встраивать в документ таблицу стилей, расположенную на сервере

Остановимся на каждом из этих способов более подробно.

Связывание. Напомним, что информация о стилях может располагаться либо в отдельном файле, либо непосредственно в коде документа. Расположение описания стилей в отдельном файле целесообразно при применении стилей при количестве страниц более 1. Для этого необходимо создать текстовый файл, описать необходимые стили и в коде документов, которые будут использовать эти стили необходимо создать ссылку на данный файл. Отметим, что данный файл может располагаться где угодно, необходимым условием является только то, чтобы броузер клиента мог его загрузить на свою сторону. Осуществляется это с помощью тега LINK, располагающегося внутри тега HEAD документов:

<LINK REL=STYLESHEET TYPE="text/css" HREF="URL">

Первые два параметра этого тега являются зарезервированными именами, требующимися для того, чтобы сообщить броузеру, что на этой страничке будет использоваться CSS. Третий параметр – HREF= «URL» – указывает на файл, который содержит описания стилей. Этот параметр должен содержать либо относительный путь к файлу – в случае, если он находится на том же сервере, что и документ, из которого к нему обращаются – или полный URL («http://...») в случае, если файл стилей находится на другом сервере.

```
<head>
<title></title>
<meta http-equiv="content-type" content="text/html; charset=windows-1251">
<link rel="stylesheet" href="css/default.css">
</head>
```

*Внедрение.*Второй вариант, при котором описание стилей располагается в коде Web-странички, внутри тега HEAD, в теге <STYLE type="text/css">... </STYLE. В этом случае вы можете использовать эти стили для элементов, располагающихся в пределах странички. Параметр type="text/css" является обязательным и служит для указания броузеру использовать CSS.

```
<head>
<style type="text/css" >
        .el_cl_1{display:inline; z-index:1};
        .el_lst{display:list-item; margin:-1%; background:#ff0000 url("bc.jpg") no-repeat};
</style>
<title></title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
</head>
```

Встраивание в теги документа. Данный, третий по счету, метод позволяет располагать описания стилей непосредственно внутри тега элемента, который описывает. Это осуществляется при помощи параметра STYLE, используемого при применении CSS с большинством стандартных тегов HTML. Данный метод нежелателен, т.к. приводит к потере одного из основных преимуществ CSS – возможности разделения информации и описания оформления информации.

```
<blockquote style="color:#CCFF66">Внимание!</blockquote>
```

Импортировании. В теге <STYLE> можно *импортировать* внешнюю таблицу стилей с помощью свойства @import таблицы стилей:

```
@import: url(styles.css);
```

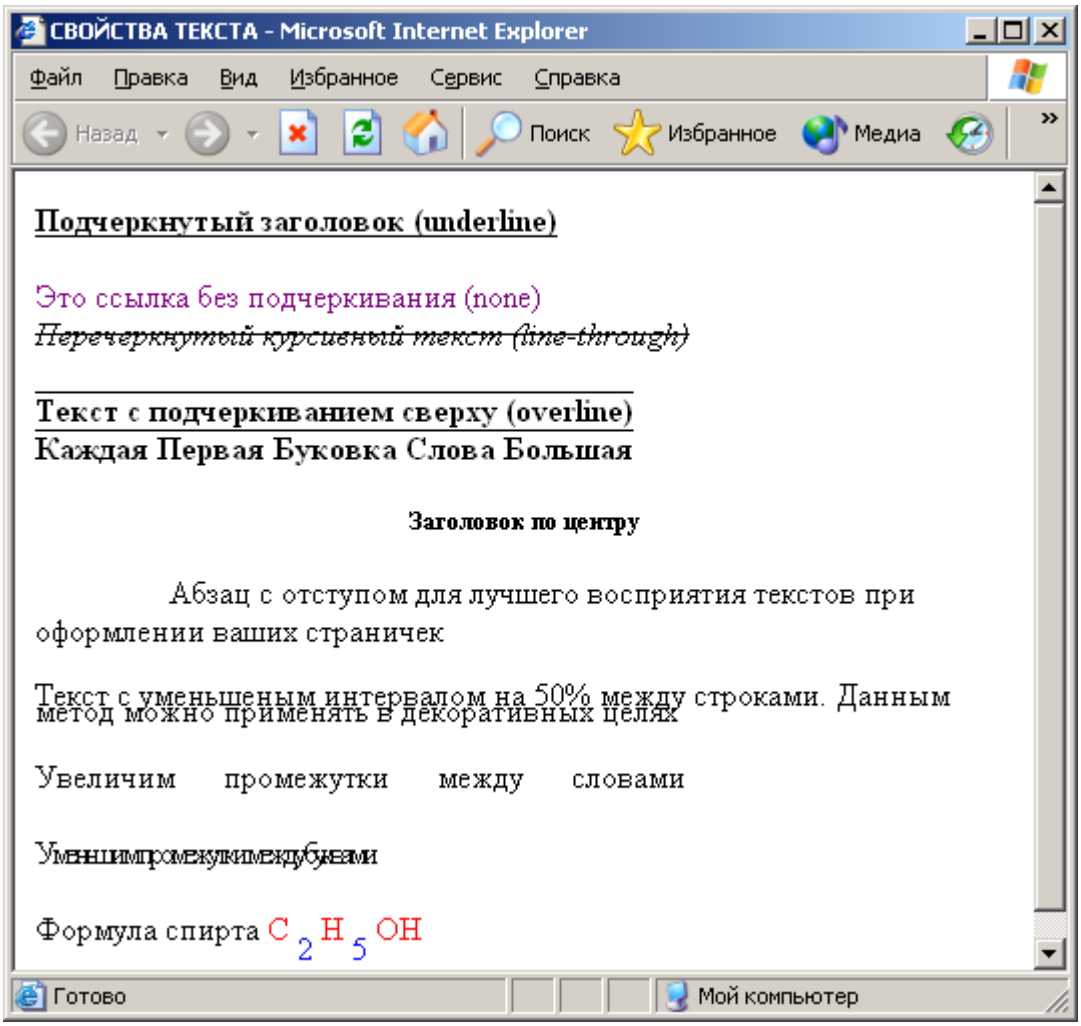
Его следует задавать в начале стилевого блока или связываемой таблицы стилей перед заданием остальных правил. Значение свойства @import является URL файла таблицы стилей.

Заметим, что импортирование от связывания отличается тем, что при

импортировании можно не только поместить внешнюю таблицу стилей в документ, но и поместить одну внешнюю таблицу стилей в другую.

```
<head>
<title>Untitled </title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251" />
<style type="text/css">
@import url('css/default.css');
</style>
</head>
```

Приведем пример использования свойства текста



Ниже приведен код примера.

```
<STYLE type="text/css">
```

```
H4 {text-decoration: underline;}
A {text-decoration: none;}
i {text-decoration:line-through;}
b {text-decoration:overline;}
H5 {text-align: center}
b.cap {text-transform:capitalize;}
.otstup {text-indent: 50pt;}
.interval {line-height: 50 %}
</STYLE>
<h4>Подчеркнутый заголовок (underline)</h4>
<a href="/css/003/text.htm">Это ссылка без подчеркивания (none)</a><br>
<i>Перечеркнутый курсивный текст (line-through)</i><p>
<b>Текст с подчеркиванием сверху (overline)</b><br>
<b class=cap>каждая первая буковка слова большая</b>
<h5>Заголовок по центру</h5>
<p class=otstup>Абзац с отступом для лучшего восприятия текстов
при оформлении ваших страничек</p>
<p class=interval>Текст с уменьшеным интервалом на 50% между строками.
Данным метод можно применять в декоративных целях</p>
<p><span style="word-spacing: 15pt">Увеличим промежутки между
словами</span>
<p><span style="letter-spacing: -2pt">Уменьшим промежутки между
буквами</span>
<p>Формула спирта
<span style=color:red>C</span>
<span style= vertical-align:sub;color:blue;>2</span>
<span style=color:red>H</span>
<span style="color:blue; vertical-align:sub;">5</span>
<span style=color:red>OH</span>
```

Задание:

Получите у преподавателя дизайн HTML-документя и оформите его при помощи CSS. Все стили необходимо вынести в отдельный файл.

Контрольные вопросы

1. Для чего используются каскадные таблицы стилей?
2. Какими способами таблицы стилей связываются с элементами документа?
3. Каковы основные отличия импортирования от связывания?
4. Каким образом сделать так, чтобы изменялся цвет ссылок только внутри тэга <b>?

Лабораторная работа №4

Тема: Программирование на PHP. PHP & MySQL

Цель работы:

Изучение технологии и получение практических навыков создания динамических web-страниц на основе данных, расположенных на сервере MySQL.

PHP. Особенности языка

PHP (англ. PHP: Hypertext Preprocessor — «PHP: препроцессор гипертекста»; первоначально Personal Home Page Tools — «Инструменты для создания персональных веб-страниц»; произносится пи-эйч-пи) — скриптовый язык программирования общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков программирования, применяющихся для создания динамических веб-сайтов.

В области программирования для сети Интернет PHP — один из популярных сценарных языков (наряду с JSP, Perl и языками, используемыми в ASP.NET) благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе лицензии PHP.

Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений. Основные из них:

- автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;
- взаимодействие с большим количеством различных систем управления базами данных (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, Microsoft SQL Server, Sybase, ODBC, mSQL, IBM DB2, Cloudscape и Apache Derby, Informix, Ovrimos SQL, Lotus Notes, DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird / InterBase, Paradox File Access, MaxDB, Интерфейс PDO);
- автоматизированная отправка HTTP-заголовков;
- работа с HTTP-авторизацией;
- работа с cookies и сессиями;

- работа с локальными и удалёнными файлами, сокетами;
- обработка файлов, загружаемых на сервер;
- работа с XForms.

В настоящее время PHP используется сотнями тысяч разработчиков. Согласно рейтингу корпорации TIOBE, базирующемся на данных поисковых систем, в июне 2013 года PHP находился на 5 месте среди языков программирования. К крупнейшим сайтам, использующим PHP, относятся Facebook, Wikipedia и др.

Входит в LAMP — распространённый набор программного обеспечения для создания и хостинга веб-сайтов (Linux, Apache, MySQL, PHP).

1. Синтаксис

Синтаксис PHP подобен синтаксису языка Си. Некоторые элементы, такие как ассоциативные массивы и цикл foreach, заимствованы из Perl.

Для работы программы не требуется описывать какие-либо переменные, используемые модули и т. п. Любая программа может начинаться непосредственно с оператора PHP. Для того, что бы интерпретатор корректно воспринимал PHP код, достаточно его заключить в специальный элемент:

<?php ваш_код **?>**

Простейшая программа Hello world на PHP выглядит следующим образом:

Пример. Приложение Hello, World!

```
<?php
    echo 'Hello, world!';
?>
```

2. Особенности языка

PHP-скрипты обычно обрабатываются интерпретатором в порядке, обеспечивающем кроссплатформенность разработанного приложения:

1. лексический анализ исходного кода и генерация лексем,
2. синтаксический анализ полученных лексем,
3. генерация байт-кода,
4. выполнение байт-кода интерпретатором (без создания исполняемого файла).

Для увеличения быстродействия приложений возможно использование специального программного обеспечения, так называемых акселераторов. Принцип их работы заключается в кэшировании однажды сгенерированного байт-кода в памяти и/или на диске, таким образом, из процесса работы приложения исключаются этапы 1—3, что в общем случае ведёт к значительному ускорению работы.

Важной особенностью является то, что разработчику нет необходимости заботиться о распределении и освобождении памяти. Ядро PHP реализует

средства для автоматического управления памятью; вся выделенная память возвращается системе после завершения работы скрипта.

3. Расширения

Интерпретатор состоит из ядра и подключаемых модулей, «расширений», представляющих собой динамические библиотеки. Расширения позволяют дополнить базовые возможности языка, предоставляя возможности для работы с базами данных, сокетами, динамической графикой, криптографическими библиотеками, документами формата PDF и тому подобным. Любой желающий может разработать своё собственное расширение и подключить его. Существует огромное количество расширений, как стандартных, так и созданных сторонними компаниями и энтузиастами, однако в стандартную поставку входит лишь несколько десятков хорошо зарекомендовавших себя. Множество расширений доступно в репозитории PECL.

4. Параметры настройки

Интерпретатор PHP имеет специальный конфигурационный файл — php.ini, содержащий множество настроек, изменение которых влияет на поведение интерпретатора. Имеется возможность отключить использование ряда функций, изменить ограничения на используемую скриптом оперативную память, время выполнения, объём загружаемых файлов, настроить логирование ошибок, работу с сессиями и почтовыми сервисами, подключить дополнительные расширения, а также многое другое. Возможно дробление большого конфигурационного файла на части. Например, широко распространена практика вынесения настроек расширений в отдельные файлы. Параметры интерпретатора могут быть переопределены в файлах конфигурации HTTP-сервера (например, .htaccess в Apache) или в самом скрипте во время выполнения при помощи команды ini_set.

Web-сервер – принцип работы

Веб-сервер — сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными.

Веб-сервером называют как программное обеспечение, выполняющее функции веб-сервера, так и непосредственно, на котором это программное обеспечение работает. В данном лабораторном курсе мы будем рассматривать программное обеспечение.

Клиент, которым обычно является веб-браузер, передаёт веб-серверу запросы на получение ресурсов, обозначенных URL-адресами. Ресурсы — это HTML-страницы, изображения, файлы, медиа-потоки или другие данные, которые необходимы клиенту. В ответ веб-сервер передаёт клиенту запрошенные данные. Этот обмен происходит по протоколу HTTP.

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов). Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.
HTTP в настоящее время повсеместно используется во Всемирной паутине для получения информации с веб-сайтов.
HTTP используется также в качестве «транспорта» для других протоколов прикладного уровня, таких как SOAP, XML-RPC, WebDAV.
Основным объектом манипуляции в HTTP является ресурс, на который указывает URI (англ. Uniform Resource Identifier) в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере файлы, но ими могут быть логические объекты или что-то абстрактное. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. (В частности для этого используется HTTP-заголовок.) Именно благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

5. Список популярных веб-серверов

На август 2011 года наиболее распространённым веб-сервером, занимающим более 65 % рынка, является Apache — свободный веб-сервер, наиболее часто используемый в UNIX-подобных операционных системах;
Некоторые другие известные веб-серверы:

- **IIS** от компании Microsoft, распространяемый с серверными ОС семейства Windows
- **nginx** — свободный веб-сервер, разрабатываемый Игорем Сысоевым с 2002 года и пользующийся большой популярностью на крупных сайтах.
- **lighttpd** — свободный веб-сервер.
- **Google Web Server** — веб-сервер, основанный на Apache и доработанный компанией Google.
- **Resin** — свободный веб-сервер приложений.
- **Cherokee** — свободный веб-сервер, управляемый только через web-интерфейс.
- **Rootage** — веб-сервер, написанный на java.

– **THTTPD** — простой, маленький, быстрый и безопасный веб-сервер.

В данный лабораторный курс не входит практикум по установке и настройке веб-серверов. Для выполнения лабораторных работ будет использоваться специальный пакет Denwer (Денвер - Джентельменский набор веб-разработчика). Данный пакет позволяет быстро устанавливать необходимый набор программных средств без углубления в процесс их конфигурирования. В зависимости т версии Денвер состоит из следующих компонентов:

1. Apache web server 2.2
2. PHP 5.2
3. MySQL Server 5.1
4. Zend Optimizer

В данном пакете предусмотрены инструменты для быстрого создания и управления локальными сайтами.

Денвер обладает большим преимуществом: Возможна установка портативной версии на USB накопитель!

Переменные и константы

Переменные в PHP представлены знаком доллара с последующим именем переменной. Имя переменной чувствительно к регистру.

Имена переменных соответствуют тем же правилам, что и остальные наименования в PHP. Правильное имя переменной должно начинаться с буквы или символа подчеркивания и состоять из букв, цифр и символов подчеркивания в любом количестве. Под буквами здесь подразумеваются символы a-z, A-Z, и байты от 127 до 255 (0x7f-0xff).

Обращение к переменным осуществляется с помощью символа $, за которым следует имя переменной.

Пример Работа с переменными

```
6.
7.  <?php
    $var = 'Bob';
    $Var = 'Joe';
    echo "$var, $Var";      // выведет "Bob, Joe"

    $4site = 'not yet';     // неверно; начинается с цифры
    $_4site = 'not yet';    // верно; начинается с символа подчеркивания
    $täyte = 'mansikka';    // верно; 'ä' это (Расширенный) ASCII 228.
    ?>
```

По умолчанию, переменные всегда присваиваются по значению. То есть, когда вы присваиваете выражение переменной, все значение оригинального выражения копируется в эту переменную. Это означает, к примеру, что, после того как одной переменной присвоено значение другой, изменение одной из них не влияет на другую.

PHP также предлагает иной способ присвоения значений переменным: присвоение по ссылке. Это означает, что новая переменная просто ссылается (иначе говоря, "становится псевдонимом" или "указывает") на оригинальную переменную. Изменения в новой переменной отражаются на оригинале, и наоборот.

Для присвоения по ссылке, просто добавьте амперсанд (&) к началу имени присваиваемой (исходной) переменной. Например, следующий фрагмент кода дважды выводит ***'Меня зовут Боб'***:

Пример Присвоение значения по ссылке

```
<?php
$foo = 'Боб';              // Присваивает $foo значение 'Боб'
$bar = &$foo;              // Ссылка на $foo через $bar.
$bar = "Меня зовут $bar";  // Изменение $bar...
echo $bar;
echo $foo;                 // меняет и $foo.
?>
```

Важно отметить, что по ссылке могут быть присвоены только именованные переменные.

Пример Передача по ссылке именованных переменных

```
<?php
$foo = 25;
$bar = &$foo;      // Это верное присвоение.
$bar = &(24 * 7);  // Неверно; ссылка на неименованное выражение.

function test()
{
  return 25;
}

$bar = &test();    // Неверно.
?>
```

8. Константы

Константы - это идентификаторы (имена) простых значений. Исходя из их названия, нетрудно понять, что их значение не может изменяться в ходе выполнения скрипта (исключения представляют "волшебные" константы, которые на самом деле не являются константами в полном смысле этого слова). Имена констант чувствительны к регистру. По принятому соглашению, имена констант всегда пишутся в верхнем регистре.

Имя константы должно соответствовать тем же правилам, что и другие имена в PHP. Правильное имя начинается с буквы или символа подчеркивания и состоит из букв, цифр и подчеркиваний.

Пример : Правильные и неправильные имена констант

```
<?php

// Правильные имена констант
define("FOO",     "something");
define("FOO2",    "something else");
define("FOO_BAR", "something more");

// Неправильные имена констант
define("2FOO",    "something");

// Это корректное объявление, но лучше его не использовать:
// PHP однажды может зарегистрировать "волшебную" константу,
// которая сломает ваш скрипт
define("__FOO__", "something");

?>
```

Типы данных

PHP является языком программирования с динамической типизацией, не требующим указания типа при объявлении переменных, равно как и самого объявления переменных. Преобразования между скалярными типами зачастую осуществляются неявно без дополнительных усилий (впрочем, PHP предоставляет широкие возможности и для явного преобразования типов).

PHP поддерживает восемь простых типов.

Четыре скалярных типа:

1) Boolean (логическое значение, ложь или истина)
2) Integer (целое число)
3) float (число с плавающей точкой, также известное как double)
4) string (строка)

Два смешанных типа:

1) array (массив)
2) object (объект)

И два специальных типа:

1) resource (ресурс)
2) NULL (нулевая константа)

Вы также можете найти несколько упоминаний типа двойной точности (double). Рассматривайте его как число с плавающей точкой, два имени существуют только по историческим причинам.

Как правило, программист не устанавливает тип переменной; обычно это делает PHP во время выполнения программы в зависимости от контекста, в котором используется переменная.

Пример: Пример работы с переменными разных типов

```
<?php
$a_bool = TRUE;   // логический
$a_str  = "foo";  // строковый
$a_str2 = 'foo';  // строковый
$an_int = 12;     // целочисленный

echo gettype($a_bool); // выводит: boolean
echo gettype($a_str);  // выводит: string

// Если это целое, увеличить на четыре
if (is_int($an_int)) {
    $an_int += 4;
}

// Если $a_bool - это строка, вывести ее
// (ничего не выводит)
if (is_string($a_bool)) {
    echo "Строка: $a_bool";
}
?>
```

Булев

Это простейший тип. boolean выражает истинность значения. Он может быть либо TRUE либо FALSE.

Синтаксис:

Для указания boolean, используйте ключевое слово TRUE или FALSE. Оба регистро-независимы.

Пример: Тип данных Булев

```
<?php
$foo = True; // присвоить $foo значение TRUE
?>
```

9. Целые числа

Integer - это число из множества ℤ = {..., -2, -1, 0, 1, 2, ...}.

Синтаксис:

Целые числа могут быть указаны в десятичной (основание 10), шестнадцатеричной (основание 16), восьмеричной (основание 8) или двоичной (основание 2) системе счисления, с необязательным предшествующим знаком (- или +).

Двоичная запись integer доступна начиная с PHP 5.4.0.

Для записи в восьмеричной системе счисления, необходимо поставить пред числом 0 (ноль). Для записи в шестнадцатеричной системе счисления, необходимо поставить перед числом 0x. Для записи в двоичной системе счисления, необходимо поставить перед числом 0b.

Пример: Целые числа

```
<?php
$a = 1234; // десятичное число
```

```
$a = -123; // отрицательное число
$a = 0123; // восьмеричное число (эквивалентно 83 в десятичной системе)
$a = 0x1A; // шестнадцатеричное число (эквивалентно 26 в десятичной системе)
?>
```

10. Строки

Строка - это набор символов, где символ - это то же самое, что и байт. Это значит, что PHP поддерживает ровно 256 различных символов, а также то, что в PHP нет встроенной поддержки Unicode. Смотрите также подробности реализации строкового типа. Строки могут быть размером более 2 Гб.

Синтаксис:

Строка может быть определена четырьмя различными способами, в данном лабораторном курсе мы рассмотрим только два из них:

- одинарными кавычками
- двойными кавычками

Одинарные кавычки

Простейший способ определить строку - это заключить ее в одинарные кавычки (символ ').

Чтобы использовать одинарную кавычку внутри строки, проэкранируйте ее обратной косой чертой (\). Если необходимо написать саму обратную косую черту, продублируйте ее (\\). Все остальные случаи применения обратной косой черты будут интерпретированы как обычные символы: это означает, что если вы попытаетесь использовать другие управляющие последовательности, такие как \r или \n, они будут выведены как есть вместо какого-либо особого поведения.

Пример: Примеры создания строк с помощью одинарных кавычек.

```
<?php
echo 'это простая строка';

echo 'Также вы можете вставлять в строки
символ новой строки вот так,
это нормально';

// Выводит: Однажды Арнольд сказал: "I'll be back"
echo 'Однажды Арнольд сказал: "I\'ll be back"';

// Выводит: Вы удалили C:\*.*?
echo 'Вы удалили C:\\*.*?';

// Выводит: Вы удалили C:\*.*?
echo 'Вы удалили C:\*.*?';

// Выводит: Это не будет развернуто: \n новая строка
echo 'Это не будет развернуто: \n новая строка';

// Выводит: Переменные $expand также $either не разворачиваются
echo 'Переменные $expand также $either не разворачиваются';
?>
```

Двойные кавычки

Если строка заключена в двойные кавычки ("), PHP распознает большее количество управляющих последовательностей для специальных символов:

Таблица – Управляющие последовательности

Последовательность	Значение
\n	новая строка (LF или 0x0A (10) в ASCII)
\r	возврат каретки (CR или 0x0D (13) в ASCII)
\t	горизонтальная табуляция (HT или 0x09 (9) в ASCII)
\v	вертикальная табуляция (VT или 0x0B (11) в ASCII) (с версии PHP 5.2.5)
\e	escape-знак (ESC или 0x1B (27) в ASCII) (с версии PHP 5.4.0)
\f	подача страницы (FF или 0x0C (12) в ASCII) (с версии PHP 5.2.5)
\\	обратная косая черта
\$	знак доллара
\"	двойная кавычка

Самым важным свойством строк в двойных кавычках является обработка переменных.

Обработка переменных

Если строка указывается в двойных кавычках, то переменные внутри нее обрабатываются.

Существует два типа синтаксиса: простой и сложный. Простой синтаксис более легок и удобен. Он дает возможность обработки переменной, значения массива (array) или свойства объекта (object) с минимумом усилий.

Сложный синтаксис может быть определен по фигурным скобкам, окружающим выражение.

Простой синтаксис

Если интерпретатор встречает знак доллара ($), он захватывает так много символов, сколько возможно, чтобы сформировать правильное имя переменной. Если вы хотите точно определить конец имени, заключайте имя переменной в фигурные скобки.

Пример: Простой синтаксис обработки переменных в строках

```
<?php
$juice = "apple";

echo "He drank some $juice juice.";
// не работает, 's' - это верный символ для имени переменной,
// но наша переменная имеет имя $juice.
```

```
echo "He drank some juice made of $juices.";
?>
```

Результат выполнения данного примера:

```
He drank some apple juice.
He drank some juice made of.
```

Сложный (фигурный) синтаксис

Он называется сложным не потому, что труден в понимании, а потому что позволяет использовать сложные выражения.

Любая скалярная переменная, элемент массива или свойство объекта, отображаемое в строку, может быть представлена в строке этим синтаксисом. Просто запишите выражение так же, как и вне строки, а затем заключите его в { и }. Поскольку { не может быть экранирован, этот синтаксис будет распознаваться только когда $ следует непосредственно за {. Используйте {\$, чтобы напечатать {$. Несколько поясняющих примеров:

Пример: Сложный синтаксис обработки переменных в строках

```
<?php
// Показываем все ошибки
error_reporting(E_ALL);

$great = 'здорово';

// Не работает, выводит: Это { здорово}
echo "Это { $great}";

// Работает, выводит: Это здорово
echo "Это {$great}";
echo "Это ${great}";
11. ?>
```

Задание

1. Скачайте и установите Denwer (www.denwer.ru)
2. Создайте файл **index.php**.
3. Определите с помощью одинарных кавычек две строковые переменные **name** и **surname**, которые соответственно содержат ваше имя и фамилию.
4. Определите с помощью двойных кавычек переменную **fullname**, данная переменная должна содержать ваше полное имя (ФИО). Данные об имени и фамилии должны быть подставлены из переменных **name** и **surname**.

5. С помощью оператора echo выведете переменную **fullname** в окно браузера.

1. Создайте файл **index.php**
2. Определит три числовые переменные **x**, **y** и **z**. Задайте им произвольное значение.
3. Определите переменную **sum**, которая будет равна результату выполнения произвольной арифметической операции над переменными **x**, **y**, **z**
4. С помощью оператора echo выведете переменные **x**, **y**, **z** и **sum** в окно браузера.

1. Создайте файл **index.php**
2. Определите три переменных: **i1** и **i2** – целые числа и **f -** число с плавающей точкой.
3. С помощью оператора **echo** поочередно выведете на экран результат попарного сложения и умножения этих переменных.

1. Создайте файл **index.php**
2. Определите две логические переменные **b1** и **b2**. Одной из них присвойте значение **ЛОЖЬ (FALSE)**, а другой **ИСТИНА (TRUE)**
3. Выясните, почему переменная со значение **ЛОЖЬ** не отображается в окне браузера.
4. С помощью оператора **echo** выведете переменные на экран

Теоретические сведения

Несомненно, одним из важных аспектов PHP является поддержка баз данных. В PHP реализована обширная поддержка практически всех существующих серверов баз данных, в том числе:

Adabas D	Informix	PostgreSQL
Dbase	Ingres	Solid
Direct MS-SQL	InterBase	Sybase
Empress	mSQL	UNIX dbm
File-Pro (read-only)	MySQL	Velods
FrontBase	ODBC	
IBM DB2	Oracle (OCI7 и OC18)	

Как показывает этот список, поддержка баз данных в PHP простирается от совместимости с базами данных, известных всем (например, Oracle), до тех, о которых многие даже не слышали. Поддержка базы данных в PHP представлена набором стандартных функций для соединения с базой, обработки запросов и разрыва связи.

Сервер MySQL дает неплохое представление об общих возможностях поддержки баз данных в PHP и, к тому же, является на данный момент

наиболее распространенным в Web. В принципе, независимо от того, с каким сервером баз данных вы будете работать, адаптация примеров не вызовет особых сложностей.

MySQL (http://www.mysql.com/) — надежная СУБД на базе SQL, разработанная и сопровождаемая фирмой T.c.X DataKonsultAB (Стокгольм, Швеция). Начиная с 1995 года, MySQL стала одной из самых распространенных СУБД в мире, что отчасти обусловлено ее скоростью, надежностью и гибкой лицензионной политикой.

Благодаря хорошим характеристикам и обширному набору стандартных интерфейсных функций, очень простых в использовании, MySQL стала самым популярным средством для работы с базами данных в PHP.

MySQL распространяется на условиях общей лицензии GNU (GPL, GNU Public License). Полное описание текущей лицензионной политики MySQL приведено на сайте MySQL (http://www.mysql.com/).

Работа с сервером MySQL в PHP

Общая последовательность действий при взаимодействии с сервером MySQL выглядит так:

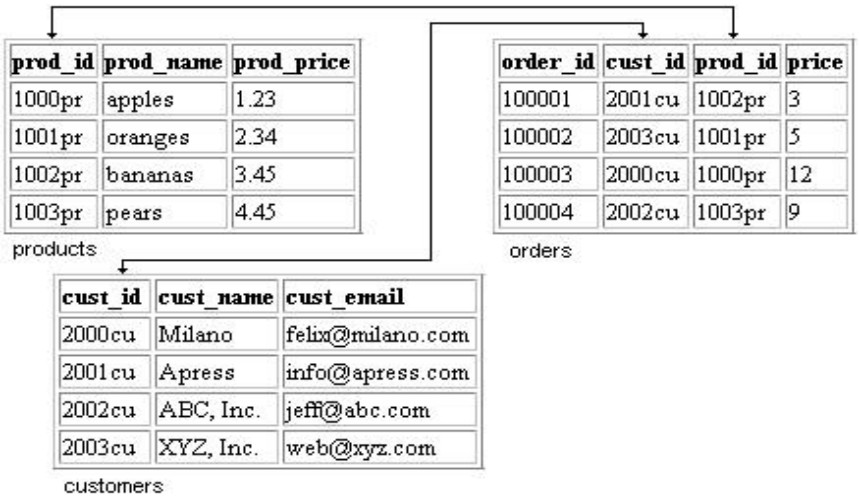
а) установить соединение с сервером MySQL. Если попытка завершается неудачей, вывести соответствующее сообщение и завершить процесс.

б) выбрать базу данных сервера MySQL. Если попытка выбора завершается неудачей, вывести соответствующее сообщение и завершить процесс. Допускается одновременное открытие нескольких баз данных для обработки запросов.

в) обработать запросы к выбранной базе (или базам).

г) после завершения обработки запросов закрыть соединение с сервером баз данных.

В примерах используются таблицы products, customers и orders.



prod_id	prod_name	prod_price
1000pr	apples	1.23
1001pr	oranges	2.34
1002pr	bananas	3.45
1003pr	pears	4.45

products

order_id	cust_id	prod_id	price
100001	2001cu	1002pr	3
100002	2003cu	1001pr	5
100003	2000cu	1000pr	12
100004	2002cu	1003pr	9

orders

cust_id	cust_name	cust_email
2000cu	Milano	felix@milano.com
2001cu	Apress	info@apress.com
2002cu	ABC, Inc.	jeff@abc.com
2003cu	XYZ, Inc.	web@xyz.com

customers

Основные функции для работы с сервером MySQL:

mysql_connect()

Функция mysql_connect( ) устанавливает связь с сервером MySQL После успешного подключения к MySQL можно переходить к выбору баз данных, обслуживаемых этим сервером. Синтаксис функции mysql_connect( ):

resource mysql_connect ([string хост [:порт] [:/путь//к/сокету] [, string имя пользователя] [, string пароль])

Возвращает: идентификатор связи MySQL при успешном выполнении, или false при ошибке.

В параметре хост передается имя хостового компьютера, указанное в таблицах привилегий сервера MySQL. Конечно, оно же используется для перенаправления запросов на web-сервер, на котором работает MySQL, поскольку к серверу MySQL можно подключаться в удаленном режиме. Наряду с именем хоста могут указываться необязательные параметры — номер порта, а также путь к сокету (для локального хоста). Параметры имя_пользователя и пароль должны соответствовать имени пользователя и паролю, заданным в таблицах привилегий MySQL. Обратите внимание: все параметры являются необязательными, поскольку таблицы привилегий можно настроить таким образом, чтобы они допускали соединение без проверки. Если параметр хост не задан, mysql_connect( ) пытается установить связь с локальным хостом.

Пример открытия соединения с MySQL:

```
@mysql_connect(" localhost", "web", "4tf9zzzf") or
                    die("Could not connect to MySQL server!");
```

В данном примере значение, возвращаемое при вызове rnysql_connect() не используется. Если в программе используется всего одно соединение с сервером MySQL, это вполне нормально. Но если программа устанавливает соединения с несколькими серверами MySQL на разных хостах, следует сохранить идентификатор соединения, возвращаемый при вызове mysql_connect( ), чтобы адресовать последующие команды нужному серверу MySQL.

Пример:

```
<?
$link1 = @mysql_connect("www.somehost.com", "web", "abcde")
                    or die("Could not connect to MySQL server!");
$linkl = @mysql_connect("www.someotherhost.com", "usr","secret")
                    or die("Could not connect to MySQL server!");
?>
```

Идентификаторы $link1 и $link2 передаются при последующих обращениях к базам данных с запросами. Вскоре вы узнаете, как именно адресовать запрос нужному серверу при помощи идентификатора соединения.

Функция **mysql_pconnect( )** обеспечивает поддержку восстанавливаемых (persistent) соединений. В многопользовательских средах рекомендуется использовать mysql_pconnect( ) вместо mysql_connect( ) для экономии системных ресурсов. По типам параметров и возвращаемого значения функция mysql_pconnect( ) в точности совпадает с mysql_connect( ).

mysql_select_db( )

После успешного соединения с MySQL необходимо выбрать базу данных, находящуюся на сервере. Для этого используется функция mysql_select_db( ). Синтаксис функции mysql_select_db( ):

bool mysql_select_db (string имя_базы_данных [,resource идентификатор_соединения])

Возвращает: true при успешном выполнении, false при ошибке Параметр имя_базы_данных определяет выбираемую базу данных, идентификатор которой возвращается функцией mysql_select_db( ). Обратите внимание: параметр идентификатор_соединения необязателен лишь при одном открытом соединении с сервером MySQL. При наличии нескольких открытых соединений этот параметр должен указываться.

Пример выбора базы данных функцией mysql_select_db( ):

```
<?
@mysql_connect("localhost", "web". "4tf9zzzf")
                    or die("Could not connect to MySQL server!");
@mysql_select_db("company")
                or die("Could not select company database!");
?>
```

mysql_close( )

После завершения работы с сервером MySQL соединение необходимо закрыть. Функция mysql_close( ) закрывает соединение, определяемое необязательным параметром. Если параметр не задан, функция mysql_close( ) закрывает последнее открытое соединение.

Синтаксис функции mysql_close( ):

bool mysql_close ([resource идентификатор_соединения])

Соединения, открытые функцией mysql_pconnect( ), закрывать не обязательно.

mysql_query( )

Функция mysql_query( ) обеспечивает интерфейс для обращения с запросами к базам данных. Синтаксис функции mysql_query( ):

resource mysql_query (string запрос [,resource идентификатор_соединения])

Параметр запрос содержит текст запроса на языке SQL. Запрос передается либо соединению, определяемому необязательным параметром идентификатор_соединения, либо, при отсутствии параметра, последнему открытому соединению.

При успешном выполнении команды SQL SELECT возвращается идентификатор результата, который впоследствии передается функции mysql_result( ) (только для SQL оператора SELECT) для последующего форматирования и отображения результатов запроса. Если обработка запроса завершилась неудачей, функция возвращает FALSE. Количество записей, участвующих в запросе, определяется при помощи функции mysql_num_rows( ). Эта функция также описана далее.

mysql_affected_rows ( )

Во многих ситуациях требуется узнать количество записей, участвующих в запросе SQL с командами INSERT, UPDATE, REPLACE или DELETE. Задача решается функцией mysql_affected_rows( ).

Синтаксис функции:

int mysql_affected_rows ([resource идентификатор_соединения])

Обратите внимание: параметр идентификатор_соединения не является обязательным. Если он не указывается, mysql_affected_rqws( ) пытается использовать последнее открытое соединение. Пример:

```
<?
// Подключиться к серверу и выбрать базу данных
@mysql_connect("localhost", "web". "4tf9zzzf")
            or die("Could not connect to MySQL server!");
  @mysql_select_db("company")
             or die("Could not select company database!");
// Создать запрос
$query = "UPDATE products SET prod_name = \"cantaloupe\" WHERE prod_id = \'10001pr\";
// Выполнить запрос
$result = mysql_query($query);
// Определить количество обновленных записей
print "Total row updated; ". mysql_affected_rows( );
mysql_close( );
?>
```

При выполнении этого фрагмента будет выведен следующий результат:

Total row updated: 1

Функция mysql_affected_rows( ) не работает с запросами, основанными на команде SELECT. Для определения количества записей, возвращенных при

вызове SELECT, используется функция mysql_num_rows( ), описанная в следующем разделе.

В одной специфической ситуации функция mysql_affected_rows( ) работает с ошибкой. При выполнении команды DELETE без секции WHEREmysql_affected_rows( ) всегда возвращает 0.

mysql_num_rows( )

Функция mysql_num_rows( ) определяет количество записей, возвращаемых командой SELECT. Синтаксис функции mysql_num_rows( ):

int mysql_num_rows(resource результат)

mysql_result( )

Функция mysql_result() используется в сочетании с mysql_query( ) (при выполнении запроса с командой SELECT) для получения набора данных. Синтаксис функции mysql_result():

mixed mysql_result (resource идентификатор_результата, int запись [. mixed поле"]')

В параметре идентификатор_результата передается значение, возвращенное функцией mysql_query( ). Параметр запись ссылается на определенную запись набора данных, определяемого параметром идентификатор_результата. Наконец, в необязательном параметре поле могут передаваться: смещение поля в таблице; имя поля; имя поля в формате имя_поля_имя_таблицы.

Пример: выборка и форматирование данных в базе данных MySQL <?

```
@mysql_connect("localhost", "web", "ffttss")
                    or die("Could not connect to MySQL server!");
@mysql_select_db("company")
                    or die("Could not select products database!");
// Выбрать все записи из таблицы products
$query = "SELECT * FROM products"; $result = mysql_query($query);
$x = 0;
print "<table>\n";
print "<tr>\n<th>Product ID</th><th>Product Name</th><th>Product Price</th>\n</tr>\n";
while ($x < mysql_numrows($result)) :
$id = mysql_result($result. $x. 'prod_id');
  $name = mysql_result($result, $x, 'prod_name');
  $price = mysql_result($result. $x, 'prod_price'); print "<tr>\n";
```

```
print "<td>$id</td>\n<td>$name</td>\n<td>$price</td>\n";
print "</tr>\n";
$x++;
endwhile;
print "</table>";
mysql_close();
?>
```

Функция mysql_result() удобна для работы с относительно небольшими наборами данных, однако существуют и другие функции, работающие намного эффективнее, — а именно, функции mysql_fetch_row() и mysql_fetch_array().

mysql_fetch_row()

Обычно гораздо удобнее сразу присвоить значения всех полей записи элементам индексируемого массива (начиная с индекса 0), нежели многократно вызывать mysql_result( ) для получения отдельных полей. Задача решается функцией mysql_fetch_row( ), имеющей следующий синтаксис:

array mysql_fetch_row (resource результат)

Использование функции list( ) в сочетании с mysql_fetch_row( ) позволяет сэкономить несколько команд, необходимых при использовании mysql_result( ).

Пример: выборка данных функцией mysql_fetch_row( )

```
<?
@mysql_connect( "localhost", "web", "ffttss")
or die("Could not connect to MySQL server!");
@mysql_select_db("company")
or die("Could not select products database!");
$query = "SELECT * FROM products";

$result = mysql_query($query);
print "<table>\n";
print "<tr>\n<th>Product ID</th><th>Product Name</th><th> Product Price</th>\n</tr>\n";
while ($row = mysql_fetch_array($result)) :
      print "<tr>\n":
      print
"<td>".$row["prod_id"]."</td>\n<td>".$row["prod_name"]."</td>\n<td>"
.$row["prod_price"]. "</td>\n";

        print "</tr>\n";
endwhile;
print "</table>";
mysql_close();
?>
```

mysql_fetch_array ()

Функция mysql_fetch_array( ) аналогична mysql_fetch_row( ), однако по умолчанию значения полей записи сохраняются в ассоциативном массиве. Можно выбрать тип индексации (ассоциативная, числовая или комбинированная). Синтаксис функции mysql_fetch_array( ):

array mysql_fetch_array (resource идентификатор результата [, тип_индексации])

В параметре идентификатор_результата передается значение,

возвращенное функцией mysql_query( ). Необязательный параметр тип_индексации принимает одно из следующих значений:

MYSQL_ASSOC — функция mysql_fetch_array( ) возвращает

ассоциативный массив. Если параметр не указан, это значение используется по умолчанию;

MYSQL_NUM — функция mysql_fetch_array( ) возвращает массив с числовой индексацией;

MYSQL_BOTH — к полям возвращаемой записи можно обращаться как по числовым, так и по ассоциативным индексам.

Задание

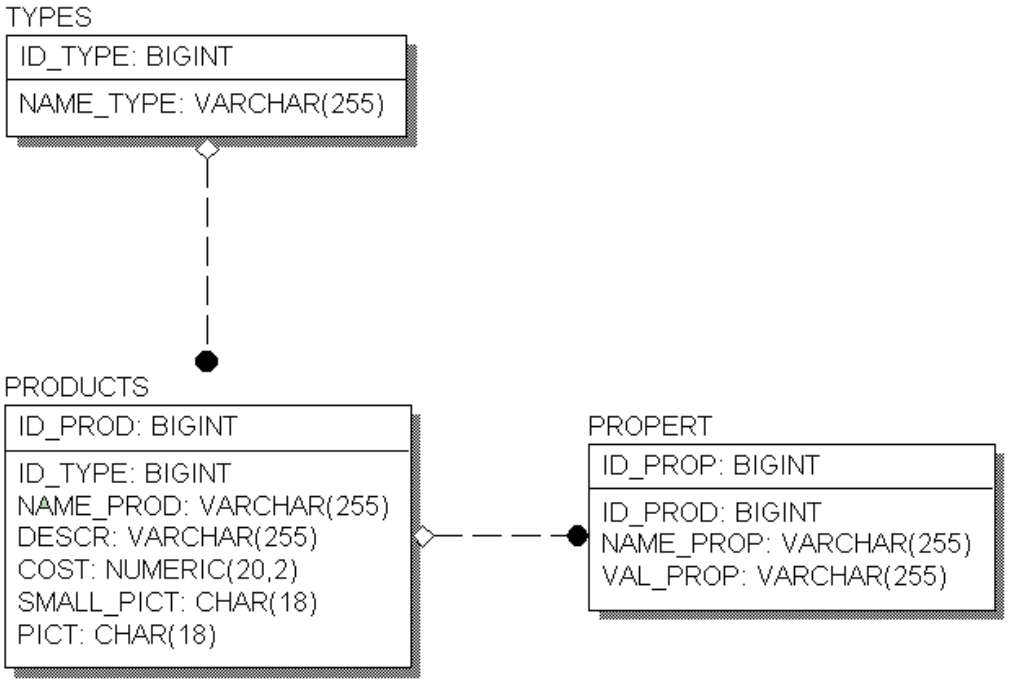
На рисунке приведенном ниже изображена структура БД каталога продукции.

Здесь таблица TYPES – содержит информацию о типах продукции или услугах. Где ID_TYPE – первичный ключ таблицы; NAME_TYPE – наименование продукции или услуг.

Таблица PRODUCTS – содержит список продукций или услуг. Первичным ключем является поле ID_PROD, вторичным ключем, т.е. ссылкой на тип продукции – ID_TYPE. Наименование продукции или услуг содержится в поле NAME_PROD, а в поле DESCR – ее описание. В поле COST – цена на ту или иную продукцию или вид услуг. Поля SMALL_PICT и PICT содержат ссылки на маленькую и большую картинки соответственно.

Таблица PROPERT содержит характеристики той или Инной продукции или вида услуг. Здесь первичным и вторичным ключами являются поля ID_PROP и ID_PROD соответственно. В поле NAME_PROP содержится информация о наименовании свойства, а в поле VAL_PROP – значение свойства.

В соответствии с данной структурой базы данных создать каталог продукции, позволяющий просматривать типы продукции или видов услуг. В соответствии с выбранным пользователем типом выводить список продукции или видов услуг, соответствующий выбранному типу. По каждому наименованию продукции или услуг предусмотреть возможность просмотра его свойств.



Контрольные вопросы

1. Что такое PHP?
2. Для чего используется PHP?
3. Какие основные особенности PHP?
4. Как определяется переменная?
5. Как определяется константа в PHP?
6. Правило именование переменных и констант.
7. Какие скалярные типы данных вы знаете?
8. Какие специальные типы данных вы знаете?
9. Какие смешанные типы данных вы знаете?
10. Тип данных: Булев.
11. Тип данных: Целое число.
12. Тип данных: Строка.
13. Чем отличаются строки, определяемые одинарными и двойными кавычками?
14. Назовите последовательность работы с БД в PHP?
15. В чем отличие методов mysql_connect и mysql_pconnect?
16. Каким образом узнать количество записей, затронутых командой UPDATE?

Лабораторная работа №5

Тема: Программирование на JavaScript

Цель:

г) ознакомление с языком составления сценариев JavaScript;

д) ознакомление с базовым синтаксисом, основными элементами JavaScript;

е) приобретение навыков создания HTML – документов с использованием JavaScript.

Сценарий

Ниже вы увидите несколько сценариев, но все они составлены по одной схеме: для каждой команды объект.свойство (object.property) создается переменная, затем переменные помещаются в команду document.write() для вывода.

Свойства объекта navigator

```
<html>
<head></head>
<body>
<script>
var an = navigator.appName;
var av = navigator.appVersion;
var acn = navigator.appCodeName;
var ua = navigator.userAgent;

document.write("Вы пользуетесь <B>" +an+ "</B>, версия " +av+ ".<BR>Кодовое название "
+acn+ ", заголовок " +ua+ "." );
</script>
</body>
</html>
```

!!! Текст в скобках должен быть весь на одной строке.

Объект navigator имеет четыре свойства. Обратите внимание на заглавные буквы!

- appName сообщает название браузера, например, Explorer.
- appVersion сообщает версию браузера и платформу, на которой он работает.
- appCodeName сообщает кодовое имя, данное браузеру, например, Netscape называет свой браузер Mozilla.
- userAgent сообщает версию используемого браузера.

Зная браузер пользователя и его версию, можно дать команду: "Если браузер такой-то, сделать то-то."

Задание 1: Создать приведенный пример скрипта в составе HTML-документа. При этом оформить операторы скрипта в виде функции,

описанную в заголовке HTML-документа, а вызов функции организовать через кнопку в теле HTML-документа.

Свойства объекта document

```
<html>
<head></head>
<body>
<script>
var bgc = document.bgColor;
var fgc = document.fgColor;
var lc = document.linkColor;
var al = document.alinkColor;
var vlc = document.vlinkColor;
var url = document.location;
var ref = document.referrer;
var t = document.title;
var lm = document.lastModified;
document.write("Цвет фона этой страницы <B>" +bgc+ "</B>.");
document.write("<BR>Цвет текста этой страницы <B>" +fgc+ "</B>.");
document.write("<BR>Цвет ссылок этой страницы <B>" +lc+ "</B>.");
document.write("<BR>Цвет активной ссылки этой страницы <B>" +al+ "</B>.");
document.write("<BR>Цвет посещенной ссылки этой страницы <B>" +vlc+ "</B>.");
document.write("<BR>URL этой страницы <B>" +url+ "</B>.");
document.write("<BR>До этого вы были на странице <B>" +ref+ "</B>.");
document.write("<BR>Заголовок этой страницы <B>" +t+ "</B>.");
document.write("<BR>Последние изменения в документ внесены: <B>" +lm+ "</B>.");
</script>
</body>
</html>
```

Код работает в IE, в других браузерах требует использования `document.bgColor.valueOf()`, а также перед выводом явного назначения свойств документа, например, `document.bgColor="#110011"`.

Обратите внимание! Цвета можно задавать через указание зарезервированных слов, например, для светлосерого `document.bgColor=LightGray`. Полная таблица цветов и их зарезервированных названия в файле: ***названия цветов в виде литералов.png***

Ниже перечислены некоторые свойства:

- bgColor возвращает шестнадцатеричный код цвет фона.
- fgColor возвращает шестнадцатеричный код цвета текста.
- linkColor возвращает шестнадцатеричный код цвета ссылки.
- alinkColor возвращает шестнадцатеричный код цвета активной ссылки.
- vlinkColor возвращает шестнадцатеричный код цвета посещенной ссылки.
- location возвращает URL страницы.
- referrer сообщает, с какой страницы пришел пользователь. Если информация недоступна, то возвращается пустое место.
- title возвращает заголовок документа HTML, т .е. текст между командами TITLE.
- lastModified сообщает дату, когда были внесены последние изменения в страницу (на самом деле дату, когда страница была загружена на сервер или сохранена последний раз на жестком диске).

- cookie (не показано) возвращает текстовый файл cookie.
- anchors (не показано) возвращает количество анкеров HREF на странице.
- forms (не показано) возвращает массив (список) объектов формы на странице.
- links (не показано) возвращает количество всех отдельных ссылок.

Задание 2: Создать приведенный пример скрипта в составе HTML-документа. При этом оформить операторы скрипта в виде функции, описанную в заголовке HTML-документа, а вызов функции организовать через кнопку в теле HTML-документа.

Индивидуальное задание 1:

Вариант 1: Используя предыдущий пример и оператор if сделайте автоматическую подстройку цветовых свойств страницы в зависимости от текущего времени. Для утренних часов сделать бежевую гамму, для дневных голубую или желтую, для вечерних серую, для ночных – синюю.

Вариант 2: Используя предыдущий пример и оператор if сделайте автоматическую подстройку цветовых свойств страницы в зависимости от текущего времени года, которое автоматически определяется по текущей дате. Для зимы сделать бело-голубую гамму, для весны – оранжевую/зеленую, для лета – зеленую/желтую, для осени – красную/серую.

Вариант 3: Используя предыдущий пример и оператор if сделайте автоматическую подстройку цветовых свойств страницы в зависимости от декады месяца. Для первой декады сделать красную/фиолетовую гамму, для второй декады - бежевую, для третьей - коричневую.

Вариант 4: Используя предыдущий пример и оператор if сделайте автоматическую подстройку цветовых свойств страницы в зависимости от дня месяца и первой или второй половины дня. Для четных дней первой половины дня сделать голубую гамму, для четных дней второй половины дня - желтую, для нечетных дней первой половины дня – зеленую, для четных дней второй половины дня – фиолетовую.

Свойства объекта history

```
<html>
<head></head>
<body>
<SCRIPT LANGUAGE="javascript">
    var h = history.length;
    document.write("До этого вы посетили " + h + " страниц.");
</SCRIPT>
</body>
</html>
```

Объект history позволяет переместиться на одну или несколько страниц вперед или назад.

Объектом является журнал посещений history. Это список страниц, которые посетил браузер во время работы. Список истории позволяет реализовать кнопку "Назад" и просмотреть еще раз любую страницу.

Также объект имеет свойство length (протяженность). Оно позволяет определить в сценарии количество страниц в папке "history".

Существует также метод go() (пойти), который позволяет передвигаться по history.length с указанным шагом.

Задание 3: Создать приведенный пример скрипта в составе HTML-документа. При этом оформить операторы скрипта в виде функции, описанную в заголовке HTML-документа, а вызов функции организовать через кнопку в теле HTML-документа.

Два свойства объекта location

```
<SCRIPT LANGUAGE="javascript">
var hst = location.host;
document.write("Страница находится на <B>" + hst + "</B>.");

var hstn = location.hostname;
document.write("Страница находится на <B>" + hstn + "</B>.");
</SCRIPT>
```

Выше представлены два свойства объекта location: host, и hostname. Команды равноценны, так как выполняют одну и ту же задачу — сообщают URL в текстовом формате или адрес IP, в зависимости от сервера. Но... location.host сообщает URL плюс "порт", с которым соединен пользователь. location.hostname сообщает только URL.

Если вы получаете одинаковый результат от обеих команд, значит, ваш сервер не соединил вас со специальным портом. Говоря техническим языком, свойство "порта" — null.

Кстати, эти две команды не работают, если просматривать страницу с жесткого диска. Результат может быть только в том случае, если она размещается на сервере, так как сценарию требуется URL для анализа.

Существует множество других свойств, с которыми вы встретитесь во время уроков. Здесь даны общие представление о свойствах — как они используются и что делают наиболее часто используемые.

Задание 4: Создать приведенный пример скрипта в составе HTML-документа. При этом оформить операторы скрипта в виде функции, описанную в заголовке HTML-документа, а вызов функции организовать через кнопку в теле HTML-документа.

Индивидуальное задание 2:

Вариант 1: Используя одну из вышеприведенных команд типа объект.свойство, напишите скрипт, который создает ссылку на страницу документа HTML на каком-либо сервере. Например, если вы находитесь на www.izi.vlsu.ru, то скрипт должен создать ссылку на www.izi.vlsu.ru/index.html. Привязать этот код к кнопке.

Вариант 2: Используя одну из вышеприведенных команд типа объект.свойство, напишите скрипт, который создает ссылку на страницу документа HTML на каком-либо сервере. Например, если вы находитесь на www.izi.vlsu.ru, то скрипт должен создать ссылку на домен www.vlsu.ru/index.html. Привязать этот код к кнопке.

Вариант 3: Используя одну из вышеприведенных команд типа объект.свойство, напишите скрипт, который создает ссылку на страницу документа HTML на каком-либо сервере и покажет номер порта. Например, если вы находитесь на www.izi.vlsu.ru, то скрипт должен создать ссылку на www.izi.vlsu.ru/index.html и отразить порт. Привязать этот код к кнопке.

Вариант 4: Используя одну из вышеприведенных команд типа объект.свойство, напишите скрипт, который создает ссылку на страницу документа HTML на каком-либо сервере и покажет номер порта. Например, если вы находитесь на www.izi.vlsu.ru, то скрипт должен создать ссылку на домен www.vlsu.ru/index.html и отразить порт. Привязать этот код к кнопке.

Также, какое бы свойство ни использовалось, присвойте ему имя переменной. Имейте в виду, что страница должна находиться на сервере, на жестком диске сценарий не работает, так как там нет никакого location.host.

Общая иерархия объектов

- Window
  - Parent
  - Self
    - Location
      - Href
    - Document
      - Image
        - Src
      - Form
        - Text
        - Submit
        - Checkbox
  - Top
  - Frames

Результат действия иерархии

Все ссылки начинаются с наивысшего объекта, window (окно браузера), и идут по нисходящей. Окна и рамки (frames) принадлежат объекту window.

На них не нужно ссылаться, если только их не больше одного. Top, self, parent и frames — "встроенные" имена для окон.

Вот несколько примеров. Обратите внимание на иерархию.

document.mypic.src = "pic1.gif"

в самом начале window не требуется. Предполагается, что это все и так находится внутри окна. Команда document.mypic.src указывает на изображение с именем mypic, и изменяет его содержимое на "pic1.gif". В данном случае document (документ) — это страница на которой находится элемент, mypic — имя элемента, а SRC — источник элемента ("pic1.gif").

document.write(location.href)

write( ) — это метод объекта document. location.href содержит полный URL окна. Обратите внимание, что location и document находятся на одном уровне. Это значит, что вы получаете адрес документа того же уровня.

Разбор иерархии объектов

- Window
  - Parent
  - Self
    - Location
      - Href
    - Document
      - Image
        - Src
      - Form
        - Text
        - Submit
        - Checkbox
  - Top
  - Frames

- Самая большая путаница в том, что некоторые объекты также являются и свойствами.
- window — только объект.
- document является свойством окна, но в свою очередь и объектом.
- form — это свойство документа, но также и объект со своими свойствами!
- value (значение) и SRC (источник) — только свойства!
- Здесь представлены не все объекты и свойства. Однако этого достаточно, чтобы понять концепцию в целом... Все ссылки начинаются сверху от window и идут по нисходящей. То есть, нельзя написать document.mytext.myform или mypic.src.document. Это неправильный порядок, следует писать слева направо от более общего к более конкретному.
- Важное замечание: чтобы показать содержимое поля формы, необходимо использовать свойство value (значение), например, document.myform.mytext.value! Если написать просто document.myform.mytext, то будет получена информация о поле формы, но не о его содержании!

Считайте value ("значение") некоторым показателем того, что нечто имеется или отсутствует в определенное время. Поле с флажком может иметь значение "on" или "off", в зависимости от того, задан он или нет. Текстовое поле может иметь значение "hidden" (скрытое), если вы не хотите, чтобы пользователь его видел. Текстовое поле, как указано выше, может содержать какую-то запись. Она будет значением этого поля.

Обработчика событий
Команды последействия: onUnLoad и onMouseOut

Это два обработчика событий, которые необходимо иметь в своем арсенале: onMouseOut и onUnload (обратите внимание на заглавные буквы). Они рассматриваются в одном уроке, потому что начинают действовать после того, как что-то сделано.

onMouseOver вызывает некое событие, если навести мышь, к примеру, на ссылку. В противоположность ей onMouseOut начинает действовать, если курсор увести со ссылки. Мы также знаем, что команда onLoad запускает сценарий, когда страница загружается. Команда onUnload действует, когда пользователь уходит со страницы.

Сценарий

Следующий код использует события при перемещении указателя мыши:

```
<html>
<head>
    <title>Пример обработки событий мыши</title>
</head>
<body>
<A HREF="index.htm" onMouseOver="window.status='Эй! Убирайся с меня!';
return true"
onMouseOut="window.status='Так-то лучше, спасибо'; return true">
Наведите курсор на эту ссылку и сместите в сторону</A>
</body>
</html>
```

Задание 5: Создать приведенный пример, если статусы не отображаются в браузере, то исправить настройки браузера так, чтобы статус окна заработал.

Индивидуальное задание 3:

Вариант 1: Используя предыдущий пример, изменить действия при срабатывании событий мыши. А именно, при наведении на надпись – изменять цветовую гамму окна в темносинюю, при уводе курсора с надписи – возвращать прежнюю цветовую гамму окна.

Вариант 2: Используя предыдущий пример, изменить действия при срабатывании событий мыши. А именно, при наведении на надпись – вывести в цикле сообщение с просьбой ввести ключевое слово, когда

ключевое слово (выбрано вами самостоятельно) будет введено, то выйти из цикла, при уводе курсора с надписи – выводить сообщение с помощью alert( ) «Курсор вне зоны надписи».

Вариант 3: Используя предыдущий пример, изменить действия при срабатывании событий мыши. А именно, при наведении на надпись – вывести в цикле сообщение с просьбой ввести результат умножения двух случайно определенных чисел в диапазоне от 1 до 10, если результат ввели правильно, то выйти из цикла, при уводе курсора с надписи – выводить сообщение с помощью alert( ) «Курсор вне зоны надписи».

Вариант 4: Используя предыдущий пример, изменить действия при срабатывании событий мыши. А именно, при наведении на надпись – вывести в цикле сообщение с просьбой ввести остаток от деления двух случайно определенных чисел в диапазоне от 1 до 10, если результат ввели правильно, то выйти из цикла, при уводе курсора с надписи – выводить сообщение с помощью alert( ) «Курсор вне зоны надписи».

Использование команды onUnload при уходе со страницы:

```
<html>
<head>
    <title>Untitled Page</title>
</head>
<BODY onUnload="alert('Уже уходите?')">

</body>
</html>
```

Задание 6: Создать приведенный пример, проверить работу.

Разбор сценария

Эффекты с мышью, как легко видеть, создаются с помощью команд onMouseOver и onMouseOut.

Обратите внимание, что эти две команды никак не связаны между собой. Вам не нужно, чтобы эти события происходили одновременно. Требуется, чтобы одно событие происходило, когда курсор мыши указывает на ссылку, а другое — когда курсор мыши смещается со ссылки. Поэтому нужно писать их как две совершенно независимые команды, каждая из которых содержит свою команду return true.

Сообщение при уходе со страницы создается с помощью команды onUnload="alert('Уже уходите?')", которая добавлена в строку BODY документа HTML. !!! Обратите внимание на двойные и одинарные кавычки. Внутри двойных — одинарные. Вторая двойная кавычка означает для браузера конец команды.

Пример

В этом задании используются onUnload, onMouseOver, и onMouseOut:

- Создайте страницу с гипертекстовой ссылкой.
- Когда курсор указывает на ссылку, в строке состояния должны появляться слова: "Привет, пользователь 'название браузера'!".
- Когда курсор уходит со ссылки, в строке состояния должен появляться текст: "Не скучаете у нас на 'URL страницы'?"
- Если щелкнуть на ссылке, должно появиться окно со словами: "Уже уходите? Сейчас всего только 'текущее время'";
- Время должно определяться с помощью функции.
- Не пользуйтесь командой onClick, чтобы вывести окно сообщения, возьмите команду onUnload.

Пример кода

```
<html>
<HEAD>
<SCRIPT>
  function goodbyedate()
  {
  var d = new Date();
  var h = d.getHours();
  var m = d.getMinutes();

  var t = h + ':' + m + ' ';

  alert
    (" Уже уходите?  Сейчас всего только " + t + ".");
  }
  </SCRIPT>
</HEAD>

<body bgcolor="FFFFCC" onUnload="goodbyedate();">

<A HREF="index.html"
  onMouseOver="window.status=
   'Привет, пользователь ' +navigator.appName;
    return true"
  onMouseOut="window.status=
   'Не скучаете у нас на ' +document.location+ '.';
    return true"> Проведите курсор мыши над этой ссылкой</A>

</BODY>

</html>
```

Задание 7: Создать приведенный пример, проверить работу. Функция, определенная в заголовке (между командами <head>) задает время. К этой функции обращается команда onUnload в строке <body> документа HTML. Переменная времени используется в команде alert. Команды onMouseOver и onMouseOut построены по той же схеме, что и в уроке, кроме переменных navigator.appName в onMouseOver и document.location в onMouseOut. Каждая из команд window.status помещена в двойные кавычки. Отрезки текста стоят в одинарных кавычках.

Открытие новых окон

window.open - в нем window (окно) — объект, а open (открыть) — метод, который на него воздействует.

```
<html>
<head>
    <title>Untitled Page</title>
</head>
<body>
<SCRIPT type="text/javascript" >

    window.open('example.html', 'Window_name1', config = 'height=300,width=300')

</SCRIPT>

</body>
</html>
```

Задание 8: Создайте скрипт, представленный выше. Выполните, обратите внимание на ошибку. Теперь создайте еще страницу с произвольным содержанием, но с названием example.html. Запустите предыдущий скрипт повторно. Проверьте, с какими параметрами откроется новое окно в браузере.

Обратите внимание! Представленный здесь сценарий будет только открывать окно. В этом окне выводится документ example.html.

Расположение на странице

Начнем с расположения сценария на странице. До сих пор всегда говорилось, что лучше помещать скрипты выше в документе, чтобы они быстрее загружались в память компьютера и начинали работать без задержки. Когда речь идет о функции, сценарий помещается между командами <HEAD>. Здесь будет сделано другое предложение.

Если вы собираетесь открывать новое окно, поместите команды, которые это делают, ближе к концу документа HTML. Проще говоря, пусть они идут в последнюю очередь. Причина простая: сначала загрузится страница, а потом откроется окно. Если команды стоят в начале, то окно появится прежде, чем пользователь увидит страницу. Скорее всего он закроет новое окно, прежде чем его можно будет использовать.

Конфигурация нового окна

Общий формат открытия окна:
('URL документа в окне', 'Название нового окна',
  config='параметры нового окна')
В рассматриваемом случае мы имеем:
('example.html', 'Window_name1', config='height=300,width=300')

- example.html— это URL страницы, которая появится в новом окне. Если страница располагается на другом сервере, то необходимо добавить http:// и так далее.
- Window_name1 — пример имени нового окна.
- config= указывает, что следующие команды относятся к конфигурации нового окна.

Команды config

Приведенные выше команды config открывают новое окно размером 300 на 300 пикселей.

Обратите внимание, что команды height (высота) и width (ширина) разделены только запятой без пробелов, а значения поставлены в одинарные кавычки, так как эти два элемента являются подкомандами config и должны выполняться совместно. Пробел для браузера означает конец команды. Ошибка.

Для команды config существует множество подкоманд. Про высоту (height) и ширину (width) мы уже знаем, они определяются в пикселях. Остальные подкоманды употребляются со словами "yes" или "no" в зависимости от того, нужны ли в новом окне эти элементы. (Можно ставить "1" вместо "yes" и "0" вместо "no", но это не обязательно.)

Помните, никаких пробелов между подкомандами и используйте одинарные кавычки. Пробел равносилен ошибке.

- toolbar= отвечает за наличие панели инструментов во вновь открытом окне. Панель инструментов в верхней части окна браузера содержит кнопки НАЗАД, ВПЕРЕД, СТОП и т.д.
- menubar= отвечает за наличие строки меню с элементами ФАЙЛ, ПРАВКА, ВИД и т.д.
- scrollbars= отвечает за наличие полосы прокрутки.
- resizable= указывает, может ли пользователь изменить размер окна по своему желанию.
- location= отвечает за наличие адресной строки во вновь открытом окне, в которой выводится URL страницы.
- directories= отвечает за наличие строки каталогов в новом окне, которая содержит закладки и т.п.
- status= отвечает за наличие строки состояния.

От строки с заголовком избавиться невозможно, хотите вы этого или нет.

Может быть, вы считаете, что все вышеперечисленное — свойства. Нет. Если вам проще их запомнить, считая свойствами, — отлично, считайте их чем угодно. Но в действительности они называются характеристиками или атрибутами. Они действуют как параметры события JavaScript.

```
<html>
<head>
    <title>Untitled Page</title>
</head>
<body>
<SCRIPT type="text/javascript" >

window.open('example.html', 'Window_name1', config =
'height=300,width=300,toolbar=1,menubar=1,scrollbars=1,resizable=1,location=1,directorie
s=1,status=1')

</SCRIPT>

</body>
```

```
</html>
```

Задание 9: Создайте скрипт, представленный выше. Проверьте, с какими параметрами откроется новое окно в браузере, если поочередно заменять значения атрибутов с 1 на 0 или использовать yes/no.

Тэги в новом окне

Например, чтобы открыть главную страницу INTUIT в основном окне, надо поместить на ней следующий код:

<A HREF="http://www.intuit.ru" TARGET="main window"></A>

Основное окно всегда имеет по умолчанию имя "main". Поэтому в команду HREF документа HTML добавляется просто команда TARGET=" " с указанием main для окна, в которое должна загрузиться страница.

А если надо, чтобы страница загрузилась в новом окне? У этого окна, как было сказано выше, тоже есть имя, здесь оно названо Window_name1. Необходимо написать просто команду ссылки HREF с указанием окна Window_name1.

Можно открыть на самом деле несколько окон, добавляя несколько команд window.open. Надо только задать окнам различные имена. Можно создавать также ссылки между окнами, указывая необходимые имена окон.

Закрытие окна

Можно создать также в документе ссылку, которая будет закрывать окно. Вот как это делается:

<A HREF="" onClick="self.close()">Щелкните, чтобы закрыть</A>

Это обычная ссылка HREF, которая никуда не ведет. Задание ссылки таким образом позволяет избежать загрузки страницы. Закрывает окно команда onClick="self.close()".

self (само, себя) — это свойство может относиться к любому объекту. В нашем случае это свойство окна. Команда close (закрыть) закрывает окно.

Открытие окна по нажатию ссылки

Допустим, что требуется открыть окно по команде, а не когда пользователь заходит на страницу. Вот как это можно сделать:

<A HREF="les11.htm" onClick="window.open('opened.html', 'joe', config='height=300,width=300')">Щелкните, чтобы открыть 'joe'</A>

Это ссылка HREF, которая направлена на саму себя. Команда onClick делает работу, а параметры содержатся в скобках ( ).

Индивидуальное задание 4:

Вариант 1: Напишите скрипт, который откроет новое окно с

характеристиками:

1. размер 400 на 400 пикселей с желтым фоном страницы и содержит две ссылки:
   - одна откроет новую страницу в главном окне;
   - вторая откроет новую страницу в текущем окне.
2. страница, которая откроется в текущем окне, должна содержать ссылку, закрывающую окно.

Вариант 2: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 200 на 300 пикселей, содержит полосу прокрутки, сделайте фон страницы зеленым.
2. открытая страница содержит ссылку на izi.vlsu.ru, которая откроется в текущем окне.

Вариант 3: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 300 на 500 пикселей, содержит адресную строку и меню браузера, сделайте фон страницы голубым.
2. открытая страница содержит ссылку на intuit.ru, которая откроется в новом окне, и на izi.vlsu.ru, которая также откроется в новом окне.

Вариант 4: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 300 на 300 пикселей, содержит меню браузера и строку состояния, сделайте фон страницы серым.
2. открытая страница содержит ссылку на intuit.ru, которая откроется в текущем окне, и ссылку, которая закроет текущее окно.

Создание и наполнение окна содержимым программно

В примере ниже создается функция, которая откроет новое окно, — причем и новое окно, и все его содержимое будет содержаться в том же документе HTML.

```
<html>
<head>
    <title>Untitled Page</title>
    <SCRIPT>
        function open1() {
            var OpenWindow = window.open('', 'newwin', config = 'height=300,width=300');
            OpenWindow.document.write("<HTML>");
            OpenWindow.document.write("<TITLE>Новое окно</TITLE>");
            OpenWindow.document.write("<BODY BGCOLOR='00ffff'>");
            OpenWindow.document.write("<CENTER>");
            OpenWindow.document.write("<font size=+1>Новое окно</font><P>");
            OpenWindow.document.write("<a href='http://www.intuit.ru' target='main'>Эта
ссылка<BR> откроется в основном окне</a><p>");
            OpenWindow.document.write("<P><HR WIDTH='60%'><P>");
            OpenWindow.document.write("<a href='' onClick='self.close()'>Эта ссылка
закроет окно</a><p>");
            OpenWindow.document.write("</CENTER>");
            OpenWindow.document.write("</HTML>");
        }
```

```
    </SCRIPT>
</head>

<body>

<button onclick="open1()">Запуск страницы в новом окне</button>

</body>
</html>
```

Задание 10: Создайте скрипт, представленный выше. Проверьте, с какими параметрами откроется новое окно в браузере.

Разбор сценария

В пример выше создаем переменную OpenWindow, под которой скрывается команда window.open(). Она выглядит следующим образом:

var OpenWindow=window.open("", "newwin", "height=300,width=300");

Формат знакомый. Единственная разница в том, что не указан URL. Пустые парные кавычки/апострофы говорят браузеру, что он должен искать в сценарии информацию о новом окне, — точно так же, как и в случае отсутствия URL в команде, которая закрывает окно. Оно бы не закрылось, если бы начала загружаться новая страница. То же самое и тут. Браузер стал бы загружать новую страницу, а не выполнять сценарий.

Теперь начинаем создавать страницу HTML, которая будет в новом окне. Вот первая строка текста:

OpenWindow.document.write("<HTML>");

Команда говорит, что строка текста должна быть записана в документ переменной OpenWindow (новое окно).

Каждая новая строка следует той же схеме. Помните: когда вы пишете HTML внутри команды document.write, вместо двойных кавычек с подкомандами ставьте одинарные. Иначе будет ошибка.

Наконец обработчик событий в кнопке по onClick вызывает созданную функцию. Можно также запустить функцию в команде BODY в методе onLoad как <body onLoad="open1()">

Индивидуальное задание 4:

Вариант 1: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 400 на 400 пикселей с желтым фоном страницы и содержит две ссылки:
   - одна откроет новую страницу в главном окне;
   - вторая откроет новую страницу в текущем окне.
2. страница, которая откроется в текущем окне, должна содержать ссылку, закрывающую окно.

Вариант 2: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 200 на 300 пикселей, содержит полосу прокрутки, сделайте фон страницы зеленым.
2. открытая страница содержит ссылку на izi.vlsu.ru, которая откроется в текущем окне.

Вариант 3: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 300 на 500 пикселей, содержит адресную строку и меню браузера, сделайте фон страницы голубым.
2. открытая страница содержит ссылку на intuit.ru, которая откроется в новом окне, и на izi.vlsu.ru, которая также откроется в новом окне.

Вариант 4: Напишите скрипт, который откроет новое окно с характеристиками:

1. размер 300 на 300 пикселей, содержит меню браузера и строку состояния, сделайте фон страницы серым.
2. открытая страница содержит ссылку на intuit.ru, которая откроется в текущем окне, и ссылку, которая закроет текущее окно.

Индивидуальное задание 5:

Вариант 1: Написать скрипт, создающий функцию, которая открывает окно. Документ, который появится в окне должен быть взять из индивидуального задания 3 по вашему варианту из лабораторной работы №3. Разумеется, добавьте еще ссылку, которая закроет окно.

Вариант 2: Написать скрипт, создающий функцию, которая открывает окно. Документ, который появится в окне должен быть взять из индивидуального задания 4 по вашему варианту из лабораторной работы №3. Разумеется, добавьте еще ссылку, которая закроет окно.

Вариант 3: Написать скрипт, создающий функцию, которая открывает окно. Документ, который появится в окне должен быть взять из индивидуального задания 2 по вашему варианту из лабораторной работы №3. Разумеется, добавьте еще ссылку, которая закроет окно.

Вариант 4: Написать скрипт, создающий функцию, которая открывает окно. Документ, который появится в окне должен быть взять из индивидуального задания 1 по вашему варианту из лабораторной работы №3. Разумеется, добавьте еще ссылку, которая закроет окно.

Работа с изображениями. Динамическое изменение изображений

В данном примере будут рассмотрены дополнительные возможности использования событий onMouseOver и onMouseOut. Как мы говорили ранее, любое событие может запускать на выполнение функцию или оператор JavaScript. Вспомните команду onLoad в теле документа HTML, которая вызывает код JavaScript из заголовка HEAD.

Представленные здесь два события происходят, когда указатель мыши перемещается на ссылку или смещается со ссылки.

Еще раз обратите внимание на то, что теги <SCRIPT> и </SCRIPT> не требуются. Обработчики событий onMouseOver и onMouseOut встраиваются в тег HTML <A HREF>. Также отметим, что для отключения вывода рамки вокруг изображения в теге <IMG SRC> включен атрибут BORDER="0".

```
<html>
<head>
    <title>Untitled Page</title>
</head>

<body>

<A HREF="http://www.intuit.ru"
onMouseOver="document.pic1.src='on.jpg'"
onMouseOut="document.pic1.src='off.jpg'">
<IMG SRC="off.jpg" BORDER=0 NAME="pic1">
</A>

</body>
</html>
```

Задание 11: Создайте пример, обратите внимание, чтобы в той же папке со скриптом были рисунки on.jpg и off.jpg. На странице выводится изображение off.jpg. Если навести на изображение указатель мыши, то изображение изменится на on.jpg. При смещении указателя мыши с изображения возвращается изображение off.jpg. Обратите внимание, что команда IMG связана с обработчиками onMouse в команде HREF через команду NAME="pic1". Это необходимо для связи команд.

Основные моменты:

1. onMouseOver и onMouseOut различают регистр. Нельзя менять заглавные и строчные буквы.
2. Так как необходимо ставить кавычки после onMouseOver= и onMouseOut=, то название файла *.jpg берется в одинарные кавычки, а не в двойные.
3. document.pic1.src следует иерархии объектов, document относится к текущему объекту документа HTML, а pic1 — это имя объекта-изображения (имя можно придумать какое угодно). src (источник) — это свойство объекта-изображения, которое указывает файл изображения.
4. В этом примере onMouseOver меняет источник изображения на on.jpg.
5. OnMouseOut заставляет объект изображение вывести off.jpg.

Динамическое изменение изображений с помощью функций

Когда нужна только одна команда JavaScript, можно включить ее в тег HTML <A HREF>. Для нескольких операторов JavaScript больше подходит функция. В реальном мире на странице часто требуется многократное изменение изображения с помощью JavaScript.

```
<HTML>
<HEAD>
  <title> Пример JavaScript </title>
<SCRIPT type="text/javascript">
    function up() {
        document.mypic.src = "on.jpg"
    }
    function down() {
        document.mypic.src = "off.jpg"
    }
</SCRIPT>
</HEAD>
  <BODY>
    <CENTER>
    <h2>Пример анимации</h2>

<A HREF="http://www.intuit.ru"
  onMouseOver="up()"  onMouseOut="down()">
<IMG SRC="off.jpg" NAME="mypic" BORDER=0></A>
  </BODY>
</HTML>
```

Задание 12: Создайте пример, обратите внимание, чтобы в той же папке со скриптом были рисунки on.jpg и off.jpg. Обратите внимание, что в сценарии созданы две функции.

```
function up() {
        document.mypic.src="up.gif"
}
function down() {
        document.mypic.src="down.gif"
}
```

Функции выполняют то же, что и команды в прошлом примере. Помните иерархию объектов? Сначала документ, потом имя, присвоенное объекту, и наконец SRC. Функции названы up() и down().

```
<HTML>
<HEAD>
  <title> Пример JavaScript </title>
<SCRIPT type="text/javascript">
function up() {
    document.mypic.src = "on.jpg"
}
function down() {
    document.mypic.src = "off.jpg"
}

function up2() {
    document.mypic2.src = "off.jpg"
}

function down2() {
    document.mypic2.src = "on.jpg"
}
```

```
</SCRIPT>
</HEAD>
  <BODY>
    <CENTER>
    <h2>Пример анимации</h2>

<A HREF="http://www.intuit.ru"
  onMouseOver="up()"  onMouseOut="down()">
<IMG SRC="off.jpg" NAME="mypic" BORDER=0></A>

<A HREF="http://www.intuit.ru"
  onMouseOver="up2()"  onMouseOut="down2()">
<IMG SRC="off.jpg" NAME="mypic2" BORDER=0></A>
  </BODY>
</HTML>
```

Задание 13: Создайте пример скрипта, выполните его. Измените его таким образом, чтобы было три ссылки в виде динамических изображений. Для первого оставьте зелено/красный круглый переключатель, для второго сделайте сине/желтый переключатель, для третьего – оранжево/ голубой переключатель.

Создание функций для обработки данных пользователя в формах

Формы всегда начинаются тегом <FORM> и заканчиваются тегом </FORM>. В этом ничего нового, простой HTML.

```
<html>
<head>
<SCRIPT type="text/javascript">
    function newcolor(color) {
        alert("Вы выбрали " + color)
        document.bgColor = color
    }
</SCRIPT>
</HEAD>
<BODY>
<p>Выберите цвет фона</p>
<FORM>
 <INPUT TYPE="button" VALUE="серый"
      onClick="newcolor('gray')">
 <INPUT TYPE="button" VALUE="светло-серый"
      onClick="newcolor('lightgray')">
</FORM>
</BODY>
</HTML>
```

Задание 14: Создайте пример скрипта, выполните его.

В скрипте при указании цвета используется литерал. Литерал является значением (VALUE), которое не изменяется. Литерал может быть числом, именем или любой случайной последовательностью чисел и имен. Помните только о том, что литерал невозможно изменить. Таким образом, следующий фрагмент сценария:

onClick="newcolor('lightgray')"

... определяет строку литерал "lightgray".

Обратите внимание: атрибут VALUE (значение) в команде INPUT не является свойством JavaScript, он выводит текст на кнопку. Он не влияет на свойства JavaScript.

Динамическое изменение содержимого форм

Будем передавать в функцию данные, которые пользователь введет в поле формы. Затем эти данные будут использованы для поиска в Google. Здесь понемногу начнем погружаться в API.

```
<html>
<head>
<SCRIPT type="text/javascript">
    function Gofindit() {
        var searchfor = document.formsearch.findthis.value;
        {
            var FullSearchUrl =
"https://www.google.ru/#hl=ru&q=" + searchfor;
            location.href = FullSearchUrl;
        }
    }

</SCRIPT>
</HEAD>
<BODY>

<FORM NAME="formsearch" action="">
Найдите в Google:
<INPUT TYPE="text" NAME="findthis" SIZE="40">
<INPUT TYPE="button" VALUE="Искать"
onClick="Gofindit()">


</FORM>

</BODY>
</HTML>
```

Задание 15: Создайте пример скрипта, выполните его. Если у вас есть доступ к Интернет, то автоматически подгрузится страница Google с результатами выполнения запроса по тем словам, которые вы введете на своей форме в поле с именем findthis.

В скрипте команда https://www.google.ru/#hl=ru&q= является фрагментом Google API для формирования простого поискового запроса. В переменную searchfor функции Gofindit() передается значение, введенное в элемент формы.

<INPUT TYPE="text" NAME="findthis" SIZE="40"> - создается элемент формы типа «поле ввода» с именем findthis и размером для ввода 40.

<INPUT TYPE="button" VALUE="Искать" onClick="Gofindit()"> - создается элемент формы типа «кнопка» с надписью Искать и при нажатии на которую будет вызвана функция Gofindit().

Разбор сценария

Этот сценарий снова требует четкого понимания иерархии объектов.

1. Во-первых, создаем функцию с переменной searchfor (искать) под названием formsearch, внутри элемента findthis (найти), который обладает свойством value (значение). Она будет результатом чего-то происходящего в объекте document.

2. Вторую функцию помещаем внутри первой. Видите вторую пару {фигурных скобок}?

3. Для второй функции создаем еще одну переменную FullSearchUrl, которая представляет собой адрес поисковой машины Google плюс значение переменной searchfor, полученное через команду document.formsearch.findthis.value.

4. Наконец, location.href приравнивается переменной FullSearchUrl. После выполнения функции пользователь попадет на итоговую страницу поиска.

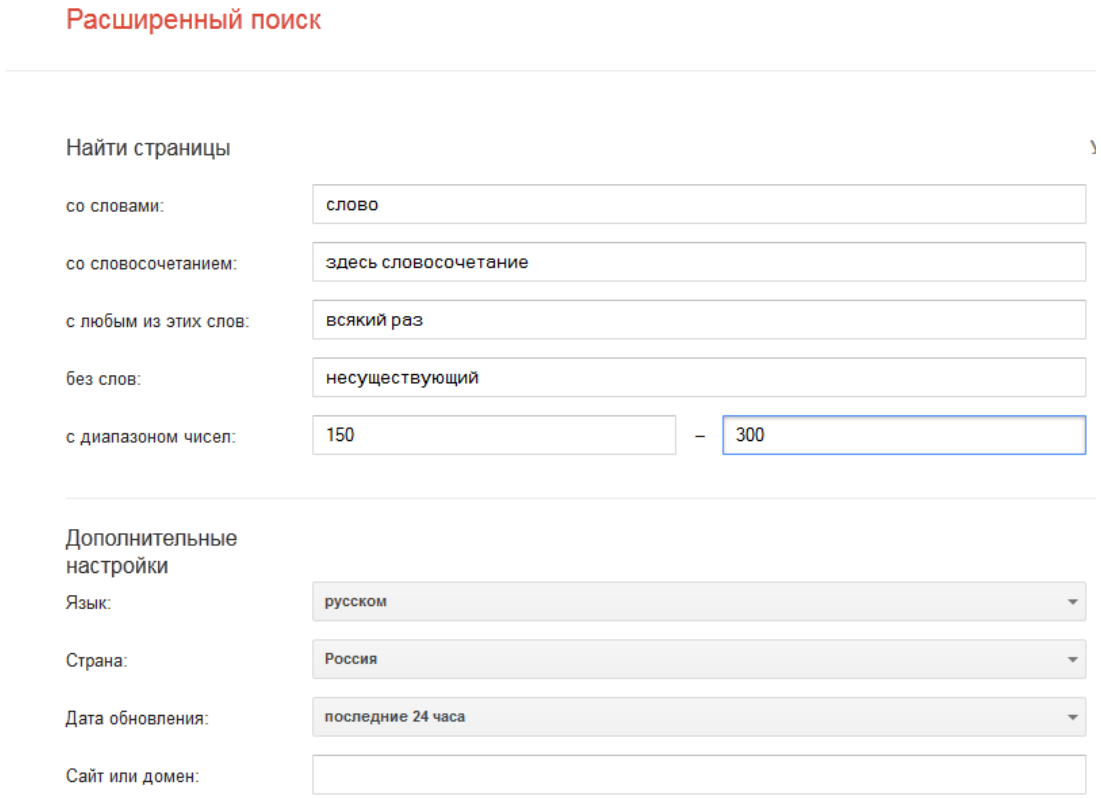
5. Теперь переходим к командам формы. Их две: текстовое поле (TEXT), куда пользователь вводит свой запрос, и кнопка, запускающая функцию.

6. Обратите внимание, что форма в целом называется formsearch. Затем для текстового поля задаем имя findthis.

7. Дальше соединяем кнопку с командой onClick, которая запускает функцию.

8. Наконец заканчиваем форму командой </FORM>. Готово.

Если в Google выбрать форму расширенного поиска, то она позволит настроить следующие поисковые элементы, например, введем следующие признаки для поиска:



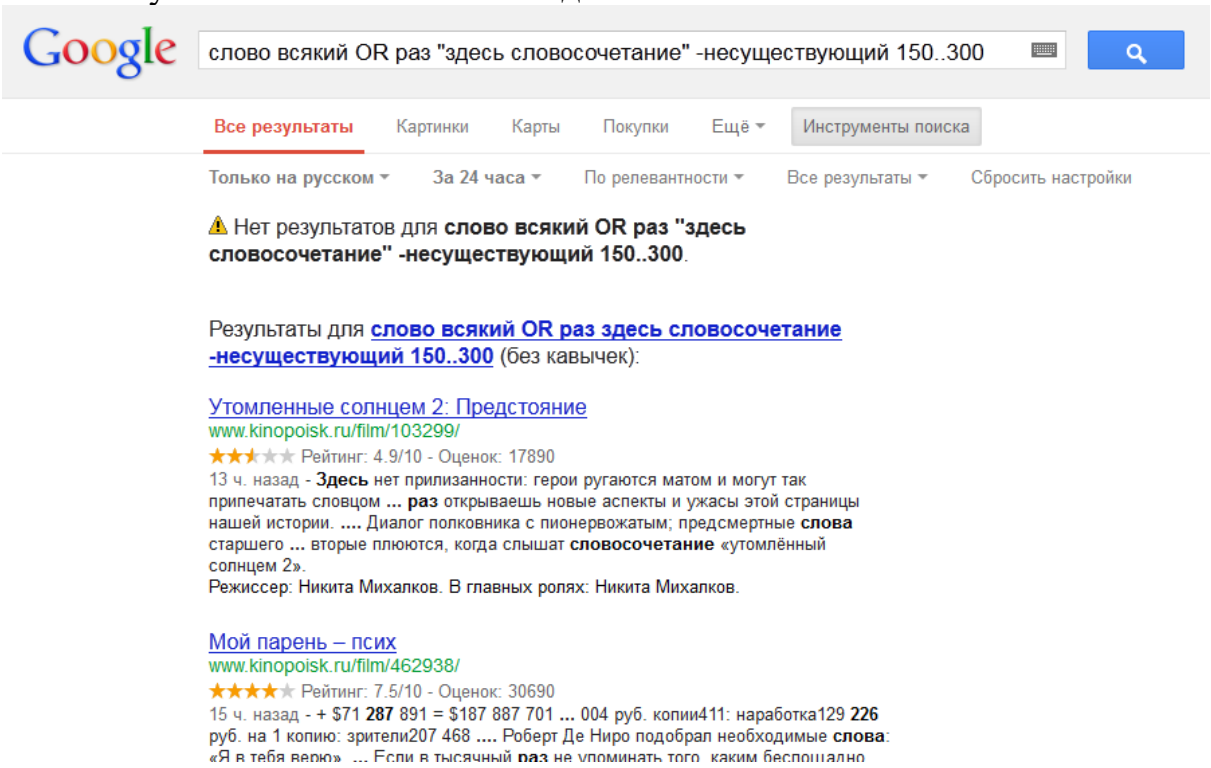
По приведенному выше примеру формы в результате будет сформирована API функция, которую можно будет внедрить в javascript:

https://www.google.com/search?hl=ru&as_q=слово&as_epq=здесь+словосочетание&as_oq=всякий+раз&as_eq=несуществующий&as_nlo=150&as_nhi=300&l

r=lang_ru&cr=countryRU&as_qdr=d&as_sitesearch=&as_occt=any&safe=images&as_filetype=&as_rights=

Если проводить анализ ключевых параметров команды, то видно, что:
& - знак присоединения следующего параметра функции;
hl=ru – язык ввода русский;
as_q= …со словами;
as_epq= … со словосочетанием;
as_oq=… с любым из этих слов;
as_eq=… без слов;
as_nlo=… с диапазоном чисел – начало;
as_nhi=… с диапазоном чисел – конец;
lr=… язык ресурсов, в которых ищем;
cr=… страна – источник ресурса, в котором ищем;
и т.д.

Результат выполнения команды ниже:



Таким образом, можно менять состав команды, удалять ненужные параметры или после знака равно ничего им не писать (что будет пониматься как игнорирование параметра), например:

https://www.google.com/search?hl=ru&as_q=слово&as_epq=здесь+словосочетание&as_oq=всякий+раз&as_eq=&as_nlo=150&as_nhi=300&lr=lang_ru&cr=countryRU&as_qdr=d&as_sitesearch=&as_occt=any&safe=images&as_filetype=&as_rights=

В результате можно создать форму, аналогичную расширенному

поиску в Google:

```
<html>
<head>
<SCRIPT type="text/javascript">
    function Gofindit() {
        var search_word = document.formsearch.search_word.value;
        var search_combination = document.formsearch.search_combination.value;
        var any_word = document.formsearch.any_word.value;
        var not_word = document.formsearch.not_word.value;
        var number_min = document.formsearch.number_min.value;
        var number_max = document.formsearch.number_max.value;
        var FullSearchUrl = "https://www.google.com/search?hl=ru&as_q=" + search_word +
                            "&as_epq=" + search_combination +
                            "&as_oq=" + any_word +
                            "&as_eq=" + not_word +
                            "&as_nlo=" + number_min +
                            "&as_nhi=" + number_max +
                            "&lr=lang_ru" +
                            "&cr=countryRU" +
                            "&as_qdr=d" +
                            "&as_sitesearch=" +
                            "&as_occt=any" +
                            "&safe=images" +
                            "&as_filetype=" +
                            "&as_rights=";

        location.href = FullSearchUrl;

    }

</SCRIPT>
</HEAD>
<BODY>

<FORM NAME="formsearch" action="">
   Найдите в Google:
   <br>
   со словом:          <INPUT TYPE="text" NAME="search_word" SIZE="40">
   <br>
   со словосочетанием: <INPUT TYPE="text" NAME="search_combination" SIZE="40">
   <br>
   с любым из слов:    <INPUT TYPE="text" NAME="any_word" SIZE="40">
   <br>
   без слов:           <INPUT TYPE="text" NAME="not_word" SIZE="40">
   <br>
   в диапазоне числа от: <INPUT TYPE="text" NAME="number_min" SIZE="40">
                      до: <INPUT TYPE="text" NAME="number_max" SIZE="40">
   <br>
   <INPUT TYPE="button" VALUE="Искать" onClick="Gofindit()">

</FORM>

</BODY>
</HTML>
```

Задание 16: Создайте пример скрипта, выполните его. Если у вас есть доступ к Интернет, то автоматически подгрузится страница Google с результатами выполнения запроса по тем словам, которые вы введете на своей форме. Обратите внимание! Выделенные желтым элементы поисковых параметров либо имеют уже введенное значение (например, lr=lang_ru), либо

не имеют вообще (например, as_sitesearch=).

Индивидуальное задание 6:

Вариант 1: Написать скрипт, который изменит предыдущий скрипт. А именно, разобраться, как указываются в расширенном поиске английский, немецкий и французский языки, добавить поле на форму, в которое пользователь будет вводить язык, а программно по условию if будет заполняться в параметре lr= соответствующий литерал (например, пользователь ввел «русский», в параметр по условию подставился литерал lang_ru).

Вариант 2: Написать скрипт, который изменит предыдущий скрипт. А именно, разобраться, как указываются в расширенном поиске страны Украина, Китай, Англия, Германия, добавить поле на форму, в которое пользователь будет вводить название страны, а программно по условию if будет заполняться в параметре cr= соответствующий литерал (например, пользователь ввел «Россия», в параметр по условию подставился литерал countryRU).

Вариант 3: Написать скрипт, который изменит предыдущий скрипт. А именно, разобраться с параметром as_occt=, добавить поле на форму, в которое пользователь будет вводить один из вариантов значений для параметра, а программно по условию if будет заполняться в параметре as_occt= соответствующий литерал.

Вариант 4: Написать скрипт, который изменит предыдущий скрипт. А именно, разобраться с параметром as_filetype=, добавить поле на форму, в которое пользователь будет вводить один из вариантов значений для параметра, а программно по условию if будет заполняться в параметре as_filetype= соответствующий литерал.

Индивидуальное задание 7:

Вариант 1: Измените сценарий задания 15 так, чтобы он вызывал другую поисковую систему - Rambler.

Вариант 2: Измените сценарий задания 15 так, чтобы он вызывал другую поисковую систему - Yahoo.

Вариант 3: Измените сценарий задания 15 так, чтобы он вызывал другую поисковую систему - Yandex.

Вариант 4: Измените сценарий задания 15 так, чтобы он вызывал другую поисковую систему - Mail.

Подсказка: чтобы увидеть, как формируется поисковая строка соответствующего поисковика, просто зайдите на их страницу напишите простой поисковый запрос. При нажатии «найти» у вас в адресной строке получится требуемая адресная строка, которую нужно будет забрать в код, но подменить конкретное значение поискового запроса на переменную, связанную с полем TEXT вашей формы.

Контрольные вопросы

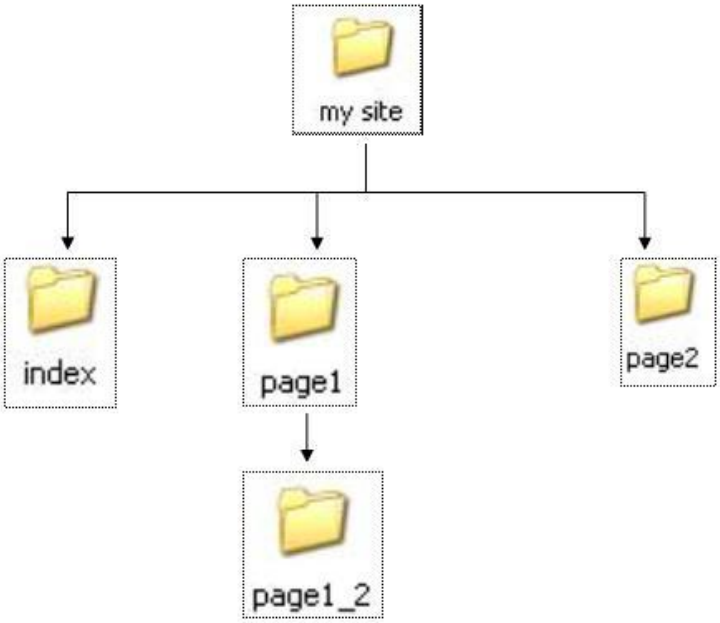
1. Каким образом JavaScrip добавляется в HTML-документ?
2. Что такое метод?
3. Каким образом объявляются объекты?
4. Назовите основные элементы, входящие в объектную модель броузера?

Лабораторная работа №6

Тема: Разработка сайта

Цель: Применив полученные ранее знания, создать Web-сайт**.**

Прежде, чем создавать web-сайт, необходимо разработать его структуру, подготовить все материалы в электронном виде в необходимом формате, создать вложенные папки внутри корневой по количеству планируемых страниц и распределить весь материал по этим вложенным папкам в соответствии с планируемым содержанием сайта. И каждую созданную страницу сохранять в соответствующую папку внутри корневой. Например, структура может быть и такой.



I. Создание первой страницы web-сайта

Создать новую страницу можно несколькими способами.

Открываем программу ***Macromedia Dreamweaver MX***

1. Воспользоваться командой **New** в **Стартовом окне**.
2. Используя команду **File\New**. Эта команда открывает диалоговое окно **New Document** (Новый документ), которое по своему составу аналогично **Стартовому окну**. На вкладке **General** (Общие) выбираем из списка **Category** (Категория) категорию создаваемого документа – **Basic page** (Основная страница). Правее откроется новый список Basic page, в котором выбираем тип страницы, а именно **HTML**. Нажимаем кнопку **Create** (**Создать**). По умолчанию эта страница будет названа **Untitled-1**. Это имя будет отображено на «закладке» у верхней границы документа.
3. В диалоговом окне **New Document** (Новый документ) из списка **Category** (Категория) выбрать пункт **Page Designs** (Дизайн страницы). В

открывшемся одновременно правее списке находится коллекция шаблонов типичных веб-страниц. Еще правее расположено окно Preview (Предварительный просмотр), в котором можно увидеть миниатюру выбранного шаблона. Нажать кнопку **Create** (Создать).

II. **Режимы отображения web-страницы**

Создание и редактирование веб-страницы можно выполнять в разных режимах отображения. Существует три режима отображения:

- Code (Код) – редактор html-кода.
- Design (Дизайн) – редактирование в визуальном режиме.
- Split (Смешанный) – смешанный режим (код и дизайн).

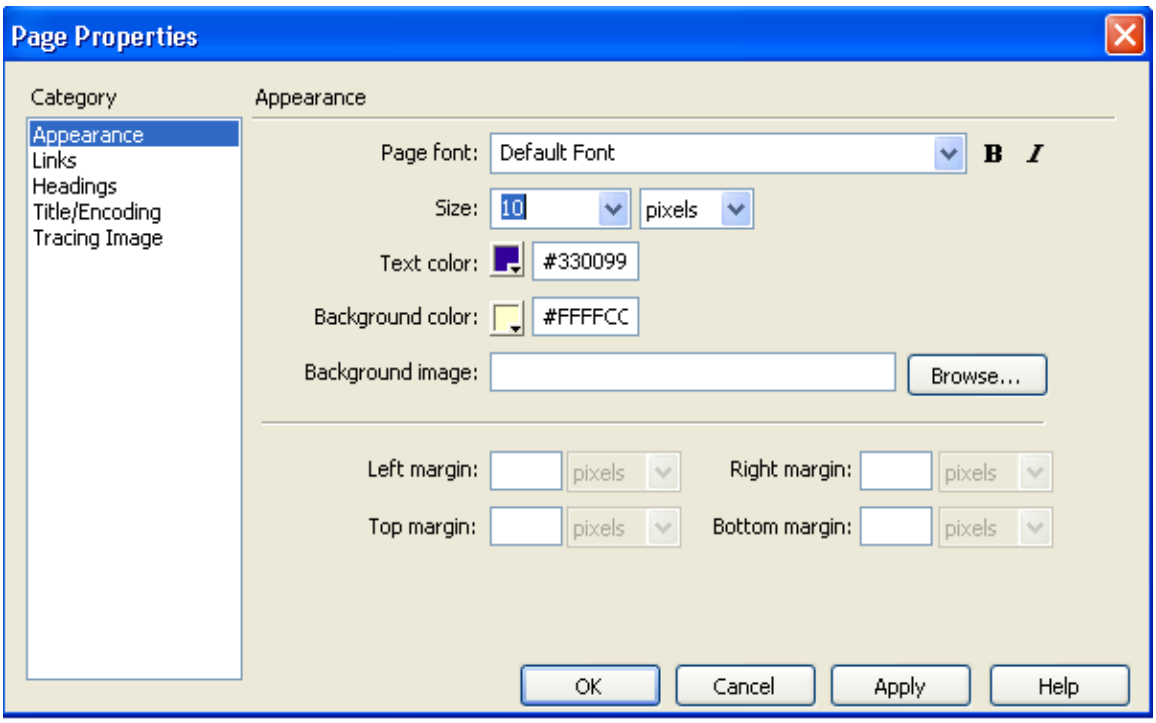
Переключение режимов выполняется нажатием соответствующих кнопок на панели инструментов документа

или выполнения соответствующей команды меню

- View/ Code (Вид/Код) – включает текстовый режим отображения (html-кода) веб-страницы.
- View/ Design (Вид/Дизайн) – включает визуальный режим отображения веб-страницы.
- View/ Code and Design (Вид/Код и Дизайн) – включает смешанный режим (html-код и дизайн) веб-страницы.

III. **Настройка свойств web-страницы**

Прежде чем оформлять еще пустую веб-страницу, необходимо настроить ее свойства. Выполните команду Modify/Page Properties (Изменить/Свойства страницы) или на панели Properties в нижней части рабочей области редактора Page Properties (Свойства страницы).



В правой части расположен список Category – здесь выбирают нужную категорию параметров веб-страницы. По умолчании открыта категория Appearance (Внешний вид). Это основная категория при настройке веб-страницы.

В раскрывающемся списке Page font (Шрифт страницы) выбирают шрифт для основного текста на странице. Разумнее всего выбирать распространенные шрифты, а не экзотические, т.к. их может не оказаться на компьютере других посетителей вашей веб-страницы в сети Интернет. Здесь же указывают стиль написания с помощью кнопок **B**- Bold (полужирный) или ***I***-Italic (курсив).

В следующем списке выбирают размер шрифта (Size) и единицы измерения этого размера, наиболее распространенный размер для web 10 пикселов.

Ниже в поле Text color (Цвет текста) выбираем цвет, которым будет окрашен основной текст.

В следующем ниже списке Background color выбирают цвет фона. Вместо однотонного фона можно использовать специально подготовленную или подобранную картинку. В поле Background image (Фоновый рисунок) указывают путь к фоновому рисунку.

Ниже следуют четыре поля для установки параметров отступа в расположении веб-страницы от краев рабочего поля, а на просмотре - все окно браузера. Чтобы веб-страница заполняла все окно браузера нужно установить нулевые значения отступов от левой, правой, верхней и нижней границы.

Возвращаемся к списку Category. Link (Ссылки) настраивается позже, когда расставлены ссылки и определен общий дизайн сайта.

Среди других в Category определим Title/Encoding (Заголовок/Кодировка). В самое верхнее поле Title вводится заголовок

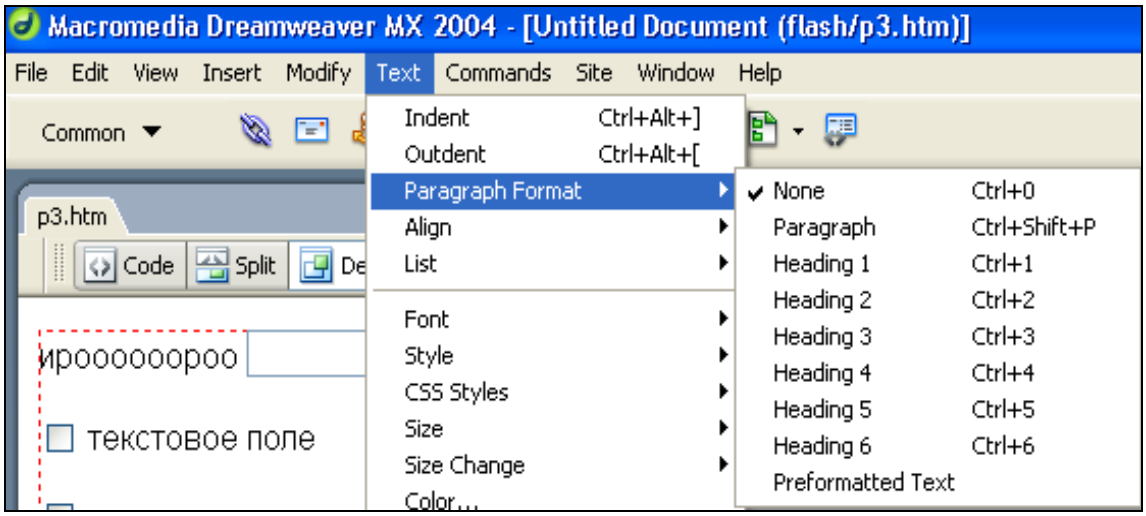
(название) веб-страницы. Следующий ниже раскрывающийся список Encoding (Кодировка) определяет очень важный момент настройки страницы – выбор кодировки. Для русского языка наиболее распространенной в настоящее время является Windows-1251. Она может еще называться Кириллица (Windows). Когда кодировка указана, нажимаем кнопку Reload (Перезагрузить).

По окончании всех установок нажимаем кнопку Ok и переходим к созданию или редактированию страницы.

IV. **Ввод и форматирование текста**

Ввод и форматирование текста можно выполнить с клавиатуры. При этом есть возможность изменения заранее заданного стиля. Изменить стиль можно с помощью панели инструментов документа. Выполните Insert/Text. Используя инструменты группы Heading (Заголовок) они помечены как h1, h2, h3 и.т.д.

Того же можно добиться используя команду меню Text/Paragraph Format (Текст/Формат абзаца). Откроется список возможных вариантов форматирования не только отдельных слов, но и абзацев. Уровней заголовков здесь уже шесть. Чем выше номер уровня, тем меньше его размер.



Пункт Paragraph (Абзац) форматирует текст как обычный абзац.

Пункт Preformatted Text (Переформатирование текста) превращает абзац в текст фиксированного формата, текст абзаца можно редактировать как в обычном текстовом редакторе.

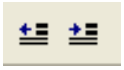
Команду меню Text/Align (Текст/Выравнивание) используют для выравнивания текста.

- Align Left – выровнять слева
- Align Center – выровнять по центру

- Align Right – выровнять справа
- Align Justify – выровнять по ширине страницы.

Для упрощения работы можно использовать панель Properties в нижней части рабочего окна. Если она закрыта, то ее можно открыть с помощью Windows/ Properties (Окно/Свойства). Для выравнивания текста служит группа кнопок

Если текст состоит их нескольких абзацев и необходимо отделить один от другого, то можно использовать клавишу Enter и/или кнопку Text Indent (задать отступ текста)



Для того чтобы снять отступ с абзаца надо нажать клавишу Enter, а затем Text Outdent (убрать отступ).

Аналогично с помощью меню

- Text/Indent
- Text/Out dent

Для **создания списков**, как и в текстовом редакторе, используем кнопки

- Unordered Lists – неупорядоченный (маркированный) список.
- Ordered Lists - упорядоченный (нумерованный) список.

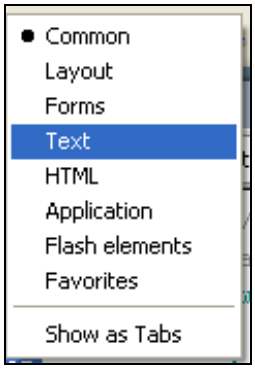
На панели инструментов Properties появится новая кнопка List Item (Пункт списка), с помощью которой можно изменить стиль значка или номер списка.

Вставку специальных символов можно осуществить так:

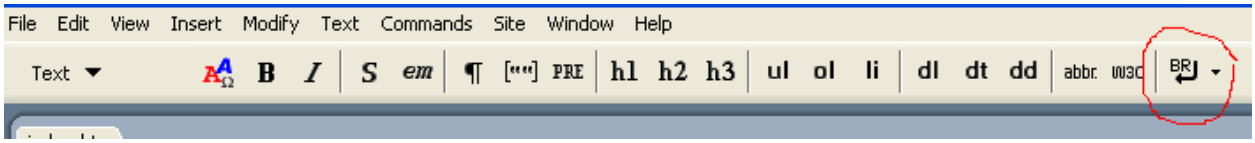
- Insert/HTML/Special Characters/Other

(Вставить/HTML/Специальный символ/Другие)

- Аналогичный результат можно получить, если выполнить следующее. На панели инструментов Insert открыть список Common в нем выбрать Text



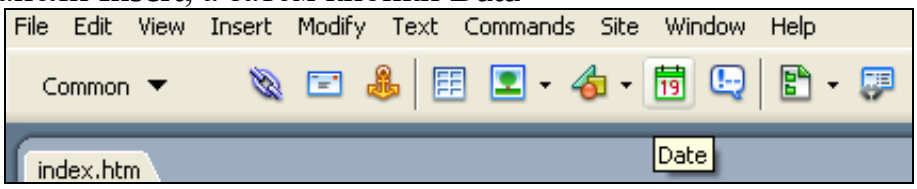
далее самую правую кнопку на этой панели



Указать дату последнего обновления сайта или страницы можно с помощью Insert/Data и далее выбрать

- Day format – формат дня недели,
- Date format – формат даты,
- Time format – формат времени.

Для автоматического сохранения даты обновления при сохранении следует установить флажок в нижней части окна Update automatically on save
Альтернативным вариантом является использование на вкладке Common (Общее) панели Insert, а затем кнопки Data



После того, как введен текст, можно изменить шрифт. Для этого требуется предварительно выделить фрагмент текста. Затем на панели Properties открываем список Font. Затем выбирают подходящую шрифтовую группу (группу похожих шрифтов).

Альтернативный вариант – выполнение команды Text/ Font (Текст/Шрифт).

Изменить размер шрифта можно с помощью списка Size на панели инструментов Properties

или

команды Text/Size (Текст/Размер), который содержит список их семи стандартных размеров шрифтов. А команда Text/Size Change (Текст/Изменить размер) содержит список величин, на которые можно изменить текущий размер.

Изменение цвета текста можно сделать с помощью панели Properties, используя селектор цвета – Text Color (Цвет текста). В правом верхнем углу над палитрой цветов расположены три кнопки (слева направо):

- Default Color сбрасывает предыдущие установки цвета текста и устанавливает цвет по умолчанию;
- System Color Picker открывает стандартное диалоговое окно выбора цвета;
- Правая кнопка (с изображением треугольника) открывает список, в котором можно выбрать режим палитры цветов. Самый последний пункт этого списка Snap to Web Safe включает так называемую безопасную веб-палитру. Цвета данной палитры должны без искажения отображаться во всех браузерах.

Альтернативным способом является команда меню Text/Color (Текст/Цвет).

Импорт текста

По мимо ввода текста с клавиатуры, его можно импортировать из других приложений с помощью команд меню Edit/Copy (Правка/Копировать) копировать текстовый фрагмент, выполненный, например, в Word, а затем Edit/Paste (Правка/Вставить) уже в *Dreamweaver*.

Полезные советы

1. Хорошим тоном считается выполнение текста на веб-страницах одного проекта одним и тем же шрифтом и одного и того же цвета.
2. Не следует для всего текста применять курсивное или полужирное выделение.
3. Не следует использовать для текста экзотические шрифты.
4. Цветовую гамму шрифт – фон нужно хорошо продумать, от этого зависит впечатление посетителя о вашем сайте.

Сохранение документа

В *Dreamweaver* имеется несколько стандартных команд меню для сохранения изменений в документе.

1. Для сохранения обновлений в уже существующей странице **File**/**Save** (Файл/Сохранить).
2. Для сохранения копий документа **File**/**Save as** (Файл/Сохранить как).
3. Для сохранения нового документа можно воспользоваться любой.
4. Для одновременного сохранения однотипных документов и последующего завершения работы **File**/**Save All** (Файл/Сохранить все). Эта команда открывает диалоговое окно **Save as** и последовательно сохраняет все документы.

V.Открытие документа

В *Dreamweaver* есть возможность редактирования веб-страниц, созданных как в этом, так и в других редакторах. Для этого веб-страницу необходимо открыть в *Dreamweaver* с помощью

1. **File**/**Open** (Файл/Открыть) или **File**/**Open Resent** (Файл/Открыть последний).
2. Стартового окна

VI. Закрытие документа

Закрытие любо документа производится стандартным образом, как и в приложениях Windows. Перед закрытием необходимо сохранить документ или редактор напомнит вам об этом.

1. **File**/**Close** (Файл/Закрыть)

2. **File/Close All** (Файл/Закрыть все) приводит к закрытию окна документа и всех открытых окон соответственно.

Контрольные вопросы

1. Для чего и как создается структура сайта?
2. Перечислите использованные инструменты при создании сайта?
3. Какие свойства сайта можно настраивать в программе *Dreamweaver ?*
4. Как настроить параметры гиперссылок?