Разработка простой web-странички

Версии HTML

Первая версия ХТМЛ была разработана в **1990-х** годах Тимом Бенерс-Ли для популярного браузера Mosaic. Но в то время применения ни для браузера ни для ХТМЛ не нашлось.

В **1993** году была разработана следующая версия HTML+, как и предыдущая, она осталась практически незаметной.

Наконец, в июне **1994** года появилась версия 2.0, которая стала популярной. Элементы этой версии используются и по сей день.

В **1994** году появилась версия 3.0, в которую были добавлены возможности прорисовки математических символов, таких знаков как: интеграл, бесконечность, дроби, фигурные скобки и т.д.

В **1996** году появилась HTML версия 3.2. Это просто революционное решение, так как в эту версию была введена специфика фреймов, которые нашли своё применение во многих сайтах. Практически все современные браузеры полностью поддерживают все элементы этой версии, поэтому авторы не сомневаются в работоспособности этой версии.

В **1997** году появилась следующая версия 4.0, называемая Dinamik HTML. Новаторство этой версии заключается в возможности делать Web-страницы интерактивными. Появились такие понятия, как JavaScript, Java, VisualBasic Script. До сих пор практически нет определенного стандарта поддержки сценариев, поэтому сколько браузеров, столько и методов поддержки скриптов. Результатом использования сценариев стали пометки в каких браузерах лучше просматривать страницы. Сейчас многим разработчикам приходится выбирать между новаторством и надежностью страницы.

Недавно большую популярность у разработчиков страниц получила технология **FLASH**. С её помощью можно создавать целые сайты, клипы, изображения, мультфильмы и даже игры и многое другое!!! Превосходства очевидны: маленький размер готового файла, интерактивность, простота в использовании, а главное, красота готового проекта.

Основы разработки HTML

Рассмотрим HTML на уровне, который позволит вам создать простейшую страничку и познакомиться с основами HTML.

Все теги записываются в угловых скобках <>. Большинство тегов имеют открывающийся элемент <> и закрывающийся </>, между которыми находится содержимое, к которому применяется этот эффект. Документ должен начинаться тегом **<HTML>** и заканчиваться закрывающим тегом **</HTML>**. Эти теги показывают, что это документ в формате HTML. Он должен содержать две части: заголовок (HEAD) и собственно документ (BODY). То есть между строчками **<HTML>** и **</HTML>** должны находится теги **<HEAD></HEAD>** и **<BODY></BODY>**. В заголовке определяется кодировка страницы (как правило KOI8-r или Windows-1251), название странички (то, которое показывается в заголовке окна) и некоторая другая информация. Для определения кодировки странички вставьте между тегами **<HEAD>** и **</HEAD>** одну из следующих строчек:

<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">

- для Windows-1251 кодировки (в этой кодировке вы можете писать код странички в

"Блокноте", используя, как русский, так и английский язык; эта кодировка наиболее предпочтительна).

<meta http-equiv="Content-Type" content="text/html; charset=koi8-r">

- для KOI8-r кодировки (как правило, эта кодировка используется визуальными HTML-редакторами, открыв документ этой кодировки в "Блокноте", вы увидите лишь какие-то какарули, однако некоторые серверы работают лишь с этой кодировкой, например сервер Chat.ru, тогда вам придется использовать специальные перекодировщики).

Для задания названия странички используются теги <TITLE></TITLE>. Между ними помещается текст названия странички.

Таким образом страничка имеет вид:

<HTML>
<HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<TITLE> Страничка Петра Петрова </TITLE>
</HEAD>
<BODY>
Содержимое странички (то, что вы должны заполнять своими тегами)
</BODY>
</HTML>

(Кстати, теги можно писать и строчными, и прописными буквами. Для браузера это безразлично. Это имеет значение лишь внутри Java-скриптов).

Задание 1. Создайте на рабочем столе папку. Задайте папке произвольное имя. Затем зайдите в эту папку. Создайте в нем пустой файл. Имя файла назовите **index.html.** Откройте этот файл. Если у вас запустился браузер, то на файле нажмите правую кнопку на манипуляторе мышь. В контекстном меню выберите «**изменить»** или «**открыть с помощью...».** Вам нужно добиться, чтобы открыть файл с помощью текстового редактора. Если вам это удалось, то вставить выделенный фрагмент из этой статьи. Сохраните его. Откройте его с помощью веб-браузера. Отметьте, что у вас получилось.

Рассмотрим основные теги, которые понадобятся вам для создания любой странички.

Начнем с текста. Большинство документов имеют заголовок. Для его создания используют теги <hx></hx>, где x – число от 1 до 6. Заключив текст между этими тегами, вы получите заголовок.

<h1>Заголовок</h1>

<h2>Заголовок</h2>

<h3>Заголовок</h3>

<h4>Заголовок</h4>

<h5>Заголовок</h5>1

<h6>Заголовок</h6>

Задание 2. Произведите исправления в **index.html.** Для этого зайдите в текстовый редактор и вставьте фрагмент выше. Не забудьте сохранится. Переключитесь на браузер и обновите страничку (это можно сделать нажав кнопку F5). Посмотрите, что получилось. Эти действия вы будете выполнять в последующих заданиях.

Для создания нового абзаца - используется тег , затем идет текст абзаца и в конце . Если вы хотите просто перейти на новую строчку без создания абзаца используйте тег
 (таким образом, тег
 не имеет закрывающегося тега).

Рассмотрим форматирование символов. Для придания тексту жирного начертания поместите его между тегами , курсивного - , подчеркнутого - <u></u>.

пример тегов	отображаемый текст
жирный текст	**жирный текст**
курсив	*курсив*
<u>подчеркнутый</u>	подчеркнутый

Некоторые теги могут применяться с атрибутами (параметрами), которые указываются в открывающемся элементе тега (можно сразу указывать несколько атрибутов в одном теге). Некоторые теги вообще не имеет смысла применять без атрибутов. К таким тегам можно отнести тег <FONT></FONT> - он может иметь несколько атрибутов (например, SIZE, FACE, COLOR), которые мы рассмотрим ниже.

Для задания размера текста используется атрибут SIZE тега <FONT>. Он позволяет задавать размер текста (по умолчанию размер текста принят равным 3). Поместив текст между тегами <FONT SIZE="x"></FONT>, где x - целое число, вы придадите ему нужный вам размер.

```
<font size="1">Поместив </font>
<font size="3">текст </font>
<font size="5">между </font>
<font size="6">тегами...</font>
```

Результат:

Поместив текст между тегами…

Задание 3. Совпадает ли ваш результат для этого фрагмента.

Задание шрифта производится с помощью атрибута FACE тега <FONT>, где FACE="Стандартный True Type шрифт". Очень аккуратно используйте этот атрибут, так как заданный шрифт должен присутствовать на компьютере пользователя - в противном случае браузер подставит шрифт, определенный по умолчанию (как правило, это Times New Roman). Применяйте шрифты, в наличии которых вы уверены, иначе пользователь увидит текст иначе, чем вы. К стандартным шрифтам, я думаю, можно отнести шрифты, поставляемые с Windows 95/98, Ms Plus, Ms Office.

Пример:

```
<font face="Times New Roman">ABC</font>

<font face="System">DEF</font>

<font face="Arial">GHI</font>

<font face="Courier">XYZ</font>
```

Результат:
ABC DEF GHI XYZ

В заключении первой части следует также сказать о возможности комментировать html-код. Комментарий помещается между <!-- и -->. Ваш комментарий может находиться в любом месте html-кода. Не бойтесь писать подробные комментарии, так как браузер пропускает их (то есть не тратит время на их загрузку). Я рекомендую вам использовать эту возможность, особенно на первых этапах изучения языка HTML.

Пример:
<!-- **Это комментарий** -->

Задание 4. Создайте комментарий. Напишите в комментарий, что делает предыдущий пример

Начнем вторую часть с добавления цветов на нашу web-страничку. Для задания цвета фона документа, цвета текста, ссылок, просмотренных (посещенных) ссылок, активных ссылок (т.е. в момент нажатия на них кнопкой мыши) используются атрибуты тега <BODY> (того самого, с которого начинается описание самой странички): BGCOLOR, TEXT, LINK, VLINK, ALINK соответственно. Для указания (определения) цвета используются именованные цвета (вынесены отдельным пунктом в разделе "HTML") или коды цветов (для этого ставится символ "#", а за ним без пробела шесть шестнадцатеричных чисел, определяющих цвет в кодировке RGB).

Пример:
<BODY BGCOLOR="#FFFF88" TEXT="#0000FF" LINK="#FF0000" VLINK="#CF2CD4" ALINK="#C033FF">

Для определения цвета конкретного фрагмента текста в документе используется атрибут COLOR тега <FONT>. Для этого также используются именованные цвета или в цвета, определяемые кодировкой RGB.

Пример:

```
<font color="#FF0000">Поместив </font>
<font color="#CF2CD4">текст </font>
<font color="#0000FF">между </font>
<font color="orange">тегами </font>
<font color="green">...</font>
```

Результат:

Поместив текст между тегами...

Для создания графического фона используется текстура - небольшая картинка (графический файл) в формате gif или jpg (jpeg). Браузер автоматически заполняет ими весь экран. Картинка должна формировать однородный фон, а с помощью небольшой полоски с плавным переходом цвета можно создавать интересные градиентные фоны. Экспериментируйте! Желательно, чтобы размер файла текстуры был небольшой. Для задания фона используется атрибут BACKGROUND тега <BODY>. В этом атрибуте вы должны указать имя файла текстуры, а при необходимости и путь, если файл текстуры не находится в той же директории, что и html-файл.

<BODY BACKGROUND="textura.gif">

А теперь рассмотрим, как вставить картинку в любое место web-странички. Для этого используются форматы gif и jpg (jpeg). Первый популярен из-за возможности использования анимации (нескольких последовательно сменяющихся кадров) и прозрачного цвета, недостаток: максимальное количество цветов 256, а формат jpg (jpeg) позволяет использовоть 24 миллионов цветов, что необходимо для фотографий, имеет хороший алгоритм сжатия, а вследствии этого небольшой размер файла.

Для вставки картинки используется тег <IMG>. Его синтаксис следующий:

<IMG SRC="имя файла" ALT="текст" ALIGN="расположение" WIDTH="ширина" HEIGHT="высота" >,

где

имя файла - это имя файла картинки с расширением. Если она находится в той же директории, что и ваша страничка, то это просто имя файла. Если картинка находится в ином месте (например где-то в Интернете), то указывайте имя с полным путем.

текст - это сообщение, которое выводится вместо картинки, если она не показывается (не найдена или пользователь настроил свой браузер так, что тот не показывает картинки). Кроме того, вы увидите этот текст в виде подсказки, когда курсор мыши находится над картинкой.

расположение - место расположения изображения. Может иметь следующие значения: **TOP** - последующий текст располагается в верхней части изображения; **BOTTOM** - последующий текст располагается в нижней части изображения; **LEFT** - изображение находится в левой части листа, текст обтекает изображение справа; **MIDDLE** - изображение находится в центре листа, **RIGHT** - изображение находится в правой части листа, текст обтекает изображение слева.

ширина - ширина изображения в пикселях.

высота- высота изображения в пикселях.

Замечание: атрибуты alt, align, width, height являются необязательными.

Например:

<img src="strelka.gif" alt="Стрелочка" width="40" height="30">

А теперь рассмотрим последнюю и, пожалуй, самую важную часть web-странички - гиперссылки.

Гиперссылки - это основа Internet'a, главный его принцип. И, соответственно, важнейшая часть web-страниц.

Гиперссылки (иногда говорят, просто ссылки) могут вести на другую страничку или сайт, на картинку (удобно не показывать большую картинку, а показать её уменьшенную копию, щелкнув на которую можно увидеть большое изображение в высоком качестве), на любой файл, на адрес электронной почты и адреса других служб WWW.

Для перехода на другую страничку используется конструкция:

<a href="URL">Название ссылки</A>, где

URL (унифицированный локатор ресурсов) - адрес любого файла в Интернете, может быть абсолютными, то есть указывается полный адрес странички (например: http://web.holm.ru/avtor.htm) или относительным, как видно из названия указывается файл относительно текущего (например avtor.htm).

Название ссылки - текст, которые будет отображаться в виде гиперссылки, например "Web.holm.ru - MEGA сайт".

Для создание ссылки на e-mail в качестве URL'a вставте "mailto:адрес электронной почты". Например, mailto:webmaster@web.holm.ru Щелкнув на такую ссылку вы откроете окно своей почтовой программы с уже записанным адресом. Останется толко написать письмо и отправить.

Часто используют не текстовую ссылку, а картинку, щелкнув на которую вы перейдете в другое место. Для этого используется следующая конструкция:

<a href="URL"><IMG SRC="картинка" width="ширина" height="высота"
border="окантовка" alt="Подсказка">

Как вы могли заметить, поскольку это гиперссылка в виде картинки, здесь появился еще один параметр: border. Он определяет толщину в пикселах обводки (окантовки). Окантовка будет выполнена тем же цветом, который определен для других гиперссылок на web-страничке. Но это не всегда смотрится красиво, поэтому зачастую значение этого параметра устанавливают в "0". Следует также сказать, что именно вышеприведенная конструкция используется при так называемом "дружеском" обмене кнопочками (баннеры размера 88x31 пиксела).

Пример:
REVERATOV.h1.ru - поиск, коллекция, анекдоты
Напишите письмо автору сайта

Замечание: изменение цвета ссылок при наведении курсора мыши у вас происходить не будет, для этого необходимо воспользоваться еще одним инструментом HTML, более подробно о котором в разделе "DHTML, CSS, SSI")

Задание 5 Выполните все выделенные фрагменты. Создайте новый файл. Назовите его как **index1.html.** Заполните его по всем правилам создания HTML. Свяжите эти странички между собой.

Заканчивая вторую часть "Основ языка HTML", заметим, что вы уже имеете базовые знания по HTML. С помощью этих немногих тегов вы уже можете творить, поэтому обдумывайте проект вашего сайта (или странички) во всемирной паутине (WWW), написанный на HTML! И удачных проектов!

Разработка Web-Странички с использованием фреймов

Используя фрэймы (frame), позволяющие разбивать web-страницы на множественные скроллируемые (или нет) подокна (как, например, на нашем сайте), в некоторых случаях вы можете значительно улучшить внешний вид и функциональность информационных систем и web-приложений. Каждое подокно (фрэйм), может иметь следующие свойства:

- Каждый фрэйм имеет свой URL, что позволяет загружать его независимо от других фрэймов
- Каждый фрэйм имеет собственное имя (параметр NAME), позволяющее обращаться к нему из другого фрэйма
- Размер фрэйма может быть изменен пользователем прямо на экране при помощи мыши (если это не запрещено указанием специального параметра)

Данные свойства фрэймов позволяют создавать продвинутые интерфейсные решения, такие как:

- Размещение статической информации, которую автор считает необходимым постоянно показывать пользователю, в одном статическом фрэйме. Это может быть графический логотип фирмы, copyright, набор управляющих кнопок
- Помещение в статическом фрэйме оглавления всех или части WEB-документов, содержащихся на WEB-сервере, что позволяет пользователю быстро находить интересующую его информацию
- Создавать окна результатов запросов, когда в одном фрэйме находится собственно запрос, а в другом результаты запроса
- Создавать формы типа "мастер-деталь" для WEB-приложений, обслуживающих базы данных

Однако, фрэйм-документ является специфичным видом HTML-документа, поскольку не содержит элемента BODY и какой-либо информационной нагрузки соответственно. Он описывает только фрэймы, которые будут содержать информацию (кроме случая двойного документа, который мы рассмотрим позже). Итак, структура файла, в котором осуществляется разбиение на фреймы, выглядит следующим образом:

```
<HTML>
<HEAD>...</HEAD>
<FRAMESET>...</FRAMESET>
</HTML>
```

ы наверняка не раз видели странички разбитые на несколько частей. Эти части и называются фреймами. Программно разбиение окна браузера на фреймы реализуется так:

1. Создается html-файл (обычно это первая страничка сервера с именем index.htm или index.html) в котором задаются размеры и количество фреймов, а также имена файлов соответствующих фреймам и некоторые атрибуты для каждого фрейма.

2. Создаются отдельные html-странички для каждого фрейма.

Попробуем создать html-файл, реализующий разбиение экрана на две части. Для этого нам понадобится два обычных html-файла, например, с именами homepage.htm и menu.htm. Главный файл назовем, к примеру, index.htm, вот как он должен выглядеть:

```
<HTML>
<HEAD>
<TITLE>Название вашей странички</TITLE>
</HEAD>
<FRAMESET cols="*,140">
<FRAME SRC="homepage.htm" NAME="frame1">
<FRAME SRC="menu.htm" NAME="frame2">
```

</FRAMESET>
</HTML>

Рассмотрим каждый тэг по отдельности:

<HTML></HTML>, <HEAD></HEAD> и <TITLE></TITLE> - стандартные тэги для всех html файлов

<FRAMESET> - в этом тэге задается количество рядов (ROWS) или столбцов (COLS), а также их размеры и расположение. Существует три способа задания размера:

1). пикселы - необходимо указать высоту или ширину в пикселах.
2). проценты - указываете сколько процентов от окна браузера вы хотите отдать фрейму и после цифр ставите знак "%" (процент). Также позаботьтесь, чтобы все ваши проценты в сумме составляли 100%.
3). звездочка - все оставшееся место в окне равняется значку *. Например, вы можете написать 20%,20%,60% или 20%,20%,* и никакой разницы не будет.

В этом же тэге можно задать толщину разграничительной линии и окаймляющей рамки командами FRAMEBORDER="X" и BORDER="Y" где x и y толщина в пикселах.

В нашем случае (<FRAMESET cols="*,140">) мы разделяем окно на два столбца, правое шириной в 140 пикселов, а левое шириной во весь оставшийся экран

<FRAME> - здесь задаются атрибуты для каждого фрейма персонально.

Команда SRC задает имя файла, который будет загружен в этом фрейме, в нашем случае имя файла homepage.htm (<FRAME SRC="homepage.htm" ...)

Команда NAME задает имя данного фрейма, в нашем случае имя "frame1" . Имя необходимо для того, чтобы в последствии можно было обратиться к этому фрейму, например загрузить в нем содержимое нового html-файла. Таким образом мы можем сделать так, чтобы нажимая на ссылку во фрейме, содержащем файл menu.htm, содержимое файла ссылки показывалось во фрейме, содержащем файл homepage.htm. Для этого нам необходимо откорректировать html-код ссылки:

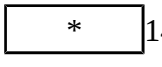
<A HREF="file.htm">file</A> - что было
<A HREF="file.htm" TARGET="frame1">file</A> - что должно быть

А если вы хотите чтобы файл загрузился в главном окне браузера (т.е. на всем пространстве окна браузера), то напишите в ссылке TARGET="_top", но имейте в виду, что после этого разбиение на фреймы исчезает.

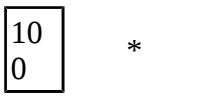
Также в этом тэге можно задать величину границы фрейма (т.е. отступ от края фрейма до его содержимого), за которую ничего кроме бэкграунда не может заходить. Это делается командами MARGINWIDTH="x" и MARGINHEIGHT="y", где x и y величина в пикселах по горизонтали и вертикали соответственно.

</FRAMESET> закрывающий тэг.

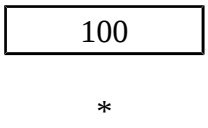
Вот несколько примеров создания фреймов:




```
<FRAMESET cols="*,140">
<FRAME SRC="homepage.htm" NAME="frame1">
<FRAME SRC="menu.htm" NAME="frame2">
</FRAMESET>
```

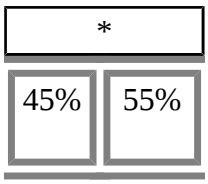



```
<FRAMESET cols="100,*">
<FRAME SRC="homepage.htm" NAME="Frame1">
<FRAME SRC="menu.htm" NAME="Frame2">
</FRAMESET>
```

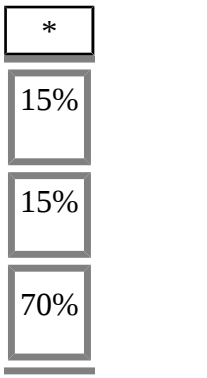



```
<FRAMESET rows="100,*">
<FRAME SRC="homepage.htm" NAME="Frame1">
<FRAME SRC="menu.htm" NAME="Frame2">
</FRAMESET>
```

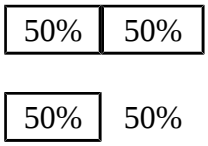

```
<FRAMESET rows="*,60">
<FRAME SRC="homepage.htm" NAME="Frame1">
<FRAME SRC="menu.htm" NAME="Frame2">
</FRAMESET>
```




```
<FRAMESET rows="*,60">
<FRAME SRC="homepage.htm" NAME="Frame1">
<FRAMESET cols="45%,55%">
<FRAME SRC="menu.htm" NAME="Frame2">
<FRAME SRC="menu2.htm" NAME="Frame3">
</FRAMESET>
</FRAMESET>
```




```
<FRAMESET cols="*,55%">
<FRAME SRC="homepage.htm" NAME="Frame1">
<FRAMESET rows="15%,15%,70%">
<FRAME SRC="menu.htm" NAME="Frame2">
<FRAME SRC="menu2.htm" NAME="Frame3">
<FRAME SRC="menu3.htm" NAME="Frame4">
</FRAMESET>
</FRAMESET>
```




```
<FRAMESET cols="50%,50%">
<FRAMESET rows="50%,50%">
<FRAME SRC="homepage.htm" NAME="Frame1">
<FRAME SRC="homepage2.htm" NAME="Frame2">
</FRAMESET>
<FRAMESET rows="50%,50%">
<FRAME SRC="menu.htm" NAME="Frame3">
<FRAME SRC="menu2.htm" NAME="Frame4">
</FRAMESET>
</FRAMESET>
```

Общий контэйнер FRAMESET описывает все фрэймы, на которые делится экран. Вы можете разделить экран на несколько вертикальных или несколько горизонтальных фрэймов. Тэг FRAME описывает каждый фрэйм в отдельности. Рассмотрим более детально каждый компонент.

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 браузеров, поддерживающих фрэймы или нет.

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. Количество подокон определяется количеством значений в списке. Общая сумма высот подокон должна составлять высоту всего окна (в любых измеряемых величинах). Отсутствие атрибута ROWS определяет один фрэйм, величиной во все окно браузера.

  Синтаксис используемых видов описания величин подокон:

  value

  Простое числовое значение определяет фиксированную высоту подокна в пикселах. Это далеко не самый лучший способ описания высоты подокна, поскольку различные браузеры имеют различный размер рабочего поля, не говоря уже о различных экранных разрешениях у пользователя. Если вы, все же, используете данный способ описания размера, то настоятельно рекомендуется сочетать его с каким-либо другим, чтобы в результате вы точно получили 100%-ное заполнение окна броузера вашего пользователя.

  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. Если указывается два или более фрэйма с описанием "*" (например "*,*"), то оставшееся пространство делится поровну между этими фрэймами. Если перед звездочкой стоит цифра, то она указывает пропорцию для данного фрэйма (во сколько раз фрейм будет больше аналогично описанного с чистой звездочкой). Например, описание "3*,*,*", говорит, что будет создано три фрэйма с размерами 3/5 свободного пространства для первого фрэйма и по 1/5 для двух других.

COLS="список-определений-горизонтальных-подокон"

- То же самое, что и ROWS, но делит окно по вертикали, а не по горизонтали.

Внимание! Совместное использование параметров ROWS и COLS может привести к непредствазуемым результатам. Например, строка: <FRAMESET ROWS="50%,50%" COLS "50%,50%"> может привести к ошибочной ситуации.

Примеры:

<FRAMESET COLS="50,*,50"> - описывает три фрэйма, два по 50 точек справа и слева, и один внутри этих полосок.

<FRAMESET ROWS="20%,3*,*"> - описывает три фрэйма, первый из которых занимает 20% площади сверху экрана, второй 3/4 оставшегося от первого фрэйма места (т.е. 60% всей площади окна), а последний 1/4 (т.е. 20% всей площади окна.

<FRAMESET ROWS="*,60%,*"> - аналогично предыдущему примеру.

Тэги <FRAMESET> могут быть вложенными, т.е. иметь следующую структуру:

<FRAMESET ROWS="50%,50%">

<FRAMESET COLS="*,*"

</FRAMESET>

</FRAMESET>

Результат данного примера мы рассмотрим позже.

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 должен быть загружен внутри данного 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. По умолчанию данное значение зависит от реализации поддержки фрэймов используемым клиентом браузером.

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. Параметр "yes" указывает, что полосы прокрутки будут в любом случае присутствовать у фрэйма, параметр "no" наоборот, что полос прокрутки не будет; "auto" определяет наличие полос прокрутки только при их необходимости (значение по умолчанию).

NORESIZE

- Данный атрибут позволяет создавать фрэймы без возможности изменения размеров. По умолчанию, размер фрэйма можно изменить при помощи мыши так же просто, как и размер окна Windows. NORESIZE отменяет данную возможность. Если у одного фрэйма установлен атрибут NORESIZE, то у соседних фрэймов тоже не может быть изменен размер со стороны данного.

NOFRAMES

Данный тэг используется в случае, если вы создаете 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.

Задание 6 Создайте проект на основе выше изложенного. Тема проекта может быть произвольной (например «мое хобби», «окружающий меня мир», «Кем я вижу себя в будушем», «Моя родословная» или любая произвольная тема), но с учетом для выполнения последующих заданий. Постройте на листочке схему перехода с одной документа на другой документ и его

связи между собой. На основе построенных схем создать необходимые файлы. Файлы расположить в следующих папках. HTML — файлы в корневом, изображения в папке img. Дополнительно необходимо создать папки css для хранения стилевых таблиц, js — для скриптов.

Немного о способе применения атрибутов

Атрибуты - часто применяемые части языка HTML. Это очень удобные конструкции, которые облегчают жизнь разработчика. Использование атрибутов лучше всего объяснить на примере элемента FONT:
** Текст **
Как вы видите - способ применения атрибутов предельно прост. Они отделяются друг от друга только пробелом. Это всегда нужно учитывать при разработке страниц и при использовании атрибутов!

Атрибуты также применяются в Стилях. Здесь синтаксис немного отличается от стандартного. Атрибуты, входящие в набор какого-либо стиля, отделяются друг от друга не пробелом, как было сказано выше, а знаком: ; [точка с запятой]. Запомните, что такое разделение применимо только к атрибутам, входящим в состав стилей.

ALIGN

Атрибут *align* используется для выравнивания объектов. Этот атрибут может принимать следующие значения:

- **left** -выравнивание по левому краю;
- **right** -выравнивание по правому краю;
- **center** -выравнивание по центру (по горизонтали);
- **middle** -выравнивание по центру (по вертикали);
- **top** -выравнивание по верхней границе;
- **bottom** -выравнивание по нижней границе;
- **justify** -выравнивание по ширине (для текста).

Для таблиц применяют атрибут *valign:*(выравнивание по вертикали)

- **top** -см. выше
- **bottom** -см. выше
- **center** -см. выше
- **baseline** -выравнивание по первой строке.

Пример:
<P align="center">

CHARSET

Атрибут применяется для определения кодировки документа.

В основном этот атрибут применяется внутри тега META.

Например:
charset="windows-1251"

КЛАССЫ (CLASS)

Этот элемент позволяет расширить возможности стилей, путем создания собственных имен стилей и последующего присвоения имени какому-либо элементу. Например,

<STYLE type="text/css">
H1.red {
color: RGB(215,40,40); text-align: center
}
</style>

Здесь мы видим, что буквам придан красивый красный цвет, а текст выровнен по центру.

Применять классы предельно просто:

<H1 class="red">текст абзаца</h1>

Прелестью классов является то, что можно создавать на одной странице несколько различных стилей(форматирований) для одного элемента.

Универсальные классы(атрибут ID)

Универсальные классы можно применять тогда, когда требуется создание класса, подходящего многим элементам сразу. Например,

#red {
color: RGB(215,40,40); font-weight:bold
}

Здесь написан стиль для полужирного начертания букв красивого красного цвета. Кстати, функция RGB незаменима при потребности в очень точном подборе цвета. Применяется этот метод при помощи атрибута ID:

<h2 id="red">Заголовок</h2>

Результатом является заголовок красного цвета полужирного начертания.

Всего должно быть в меру! Обилие стилей ухудшает восприятие, а страница выглядит новогодней ёлкой!

CLASSID

Этот атрибут задает программу или объект, которые могут быть использованы в определенных элементах.

COLOR

Атрибут *color* используется для предания определенного оттенка объектам. Использование этого атрибута очень простое. Вот пример:

color="цвет"

Можно применять в качестве цвета английское название, а можно RGB оттенок.

Некоторые примеры цветов и оттенков смотрите здесь.

Например:
<P color="red">абзац красного цвета</p>

DIR

в некоторых языках буквы читаются справа на лево. Для этого существует атрибут dir. Он может принимать значения:

dir="LTR" -слева направо
dir="RTL" -справа налево

Этот атрибут теоретически может применяться в разных элементах, хотя некоторые браузеры не поддерживают атрибут. Есть более надежный способ при помощи элемента BDO

ID

Этот атрибут применяется для расширения возможностей разработчика. Он является универсальным именем элемента. В зависимости от типа элемента, этот атрибут выполняет различные функции. Атрибут позволяет использовать классы(следующая ссылка в меню). Например, надо придать уникальный цвет, размер, шрифт и т.д. для кокой-либо части страницы. Придадим буквам серо-стальной цвет. Назовем его « rus &raquo. Для чего, внутри элемента style вводите таблицу стилей:

#rus {
color: RGB(155,180,190); font-weight: bold
}

Из-за того, что оттенок подбирается очень точно без функции RGB() не обойтись. Буквы будут очень светлыми, поэтому придадим им полужирное начертание. Этот стиль можно будет применять к различным элементам, например:

<H2 id="rus"> Заголовок, отформатированный стилем "rus"</h2>

Такой абзац будет выглядеть очень оригинально, так как будет отличаться от других. Так можно менять обилие установок по умолчанию.

Но не забывайте, что всего должно быть в меру! И излишняя разукрашенность может притупить восприятие страницы!

LANG

Атрибут определяет, на каком языке написан текст внутри элемента. Хотя этот атрибут применяется очень редко. Его даже можно назвать устаревшим.
lang= "код языка"

Вот пример основных кодов: **ar** -арабский
de -немецкий
el -греческий
en -английский
eu-us -американская версия английского языка
es -испанский
fr -французский
he -иврит
hi -хинди
it -итальянский
ja -японский
nl -голландский
pt -португальский
ru -русский

zh -китайский

но в некоторых языках буквы читаются справа на лево. Для этого существует атрибут dir.

Пример:
<FONT lang="en">Text

SIZE

Этот атрибут применяют для указания желаемого размера шрифта. Он применяется внутри элементов BASEFONT, FONT Атрибут принимает размеры от 1 до 7, но можно указать не точную цифру, а прибавление или вычитание из величины по умолчанию(по умолчанию используется 3-й размер). Пример использования:

size="число"
size="+число(до 4)"
size= "-число(до 2)"

Вот визуальные примеры использования атрибута:
Шрифт 1
Шрифт 2
Шрифт 3
Шрифт 4
Шрифт 5
Шрифт 6
Шрифт 7

Шрифт -1
Шрифт -2

Шрифт +1
Шрифт +2
Шрифт +3
Шрифт +4

TITLE

Этот атрибут в отличие от longdesc применяют к элементу для короткого комментария, который появляется на экране в виде всплывающей строки.

title="маленький комментарий"

Например:
<P title="Комментарий к абзацу">Абзац</p>

TYPE

Атрибут *type* определяет тип документа, который используется в ссылке. Первоначально *type* применялся для определения формата отправленной электронной почты, но сеанс служит для указания форматов документов, в составе Web -страниц. Например:

- **text/plain** -обычный текст;
- **text/css** -каскадная таблица стилей;
- **text/html** -текст в формате HTML;
- **application/postscript** -документ в формате PostScript;
- **application/java** -апплет;
- **image/gif; image/jpg; image/png** -изображения в формате GIF, JPG и PNG, соответственно;
- **video/mpeg** -видеоролик;
- **text/javascript** --программа-сценарий на JavaScript;
- **text/vbscript** -программа-сценарий на VBScript.

Для форм атрибут имеет другой смысл: определенного элемента формы(кнопки, поле ввода и т.д.)

Например:
<STYLE type="text/css">Стиль</style>

STYLE

Применение атрибута *style*:

<P style="свойство1: значение; свойство2: значение; ...">

Например:
<P style="font-size: 14pt; font-style: italic; color: red">

Здесь для абзаца выбран 14-й размер букв, курсив и красный цвет букв.

Так выглядит абзац, отформатированный стилем, указанном в примере.

CLEAR

Атрибут *clear* позволяет выравнивать объекты, например рисунки, относительно текста, в котором использован элемент *BR*.
Значения атрибута:

- **left** -выравнивание по левому краю
- **right** -выравнивание по правому краю
- **center** -выравнивание по центру

Например:
<BR clear="center">

FACE

Этот атрибут применяется вместе с элементом FONT. Он задает один или несколько шрифтов (через точку с запятой)

Например,
face="arial; verdana; tahoma"

К сожалению есть проблема: Так как Web -страницы просматривают много людей, то нет гарантии, что отобразится нужный шрифт на всех компьютерах. Если на компьютере нет шрифта из списка, то страница будет просматриваться с шрифтом по умолчанию.

HEIGHT

Атрибут выражает высоту какого-либо объекта. Этот атрибут обычно используется для задания высоты изображения.

height=число

WIDTH

Атрибут выражает ширину какого-либо объекта. Этот атрибут обычно используется для задания ширины изображения.

width=число

Число может принимать ряд значений: в пикселях, в процентах и т.д.

HSPACE

Атрибут определяет пустое пространство **справа и слева** от объекта и употребляется:

hspace="число(в пикселях)"

VSPACE

Атрибут определяет пустое пространство **сверху и снизу** от объекта и употребляется:

vspace="число(в пикселях)"

Задание7. Примените атрибуты к имеющимся в ваших файлах тегам и посмотрите, как изменяется поведение вашего тега. Для сравнения можете дублировать теги с атрибутами и без них.

Использование таблиц

TABLE

(HTML 3.2) – Table

Элемент для создания таблицы. Обязательно должен иметь начальный и конечный теги. По умолчанию таблица печатается без рамки, а разметка осуществляется автоматически в зависимости от объема содержащейся в ней информации. Ячейки внутри таблицы создаются с помощью элементов `TR`, `CAPTION`, `TD` и `TH`.

Атрибуты

`ALIGN` – определяет способ горизонтального выравнивания таблицы. Возможные значения: `left`, `center`, `right`. Значение по умолчанию – `left`.

`VALIGN` – должен определять способ вертикального выравнивания таблицы. Возможные

значения: top, bottom, middle.

BORDER – определяет ширину внешней рамки таблицы (в пикселах). При BORDER="0" или при отсутствии этого атрибута рамка отображаться не будет.

CELLPADDING – определяет расстояние (в пикселах) между рамкой каждой ячейки таблицы и содержащимся в ней материалом.

CELLSPACING – определяет расстояние (в пикселах) между границами соседних ячеек.

WIDTH – определяет ширину таблицы. Ширина задается либо в пикселах, либо в процентном отношении к ширине окна браузера. По умолчанию этот атрибут определяется автоматически в зависимости от объема содержащегося в таблице материала.

HEIGHT – определяет высоту таблицы. Высота задается либо в пикселах, либо в процентном отношении к высоте окна браузера. По умолчанию этот атрибут определяется автоматически в зависимости от объема содержащегося в таблице материала.

BGCOLOR – определяет цвет фона ячеек таблицы. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов.

BACKGROUND – позволяет заполнить фон таблицы рисунком. В качестве значения необходимо указать URL рисунка.

Пример 1:

```
<TABLE BORDER>
<TR>
<TH ROWSPAN=2>HDD</TH>
<TD>WD Caviar 3.1Gb</TD><TD ALIGN="right">85$</TD>
</TR>
<TR>
<TD>Quantum FB ST 6.4Gb</TD><TD ALIGN="right">110$</TD>
</TR>
</TABLE>
```

Пример 2:

```
<TABLE BORDER>
<TR>
<TH COLSPAN=2>Video</TH>
</TR>
<TR>
<TD>Matrox G400</TD><TD ALIGN="right">115$</TD>
</TR>
<TR>
<TD>Voodoo III</TD><TD ALIGN="right">129.95$</TD>
</TR>
</TABLE>
```

Задание 8. Вставте в HTML приведенные примеры и посмотрите что у вас получилось.

CAPTION

(HTML 3.2) - Caption

Задает заголовок таблицы. Содержание заголовка должно состоять только из текста. Использование блочных элементов в этом случае недопустимо.

Атрибуты

ALIGN - определяет способ вертикального выравнивания заголовка таблицы. Возможные значения:

- `top` - помещает заголовок над таблицей (значение по умолчанию);
- `bottom` - помещает заголовок под таблицей.

TR

(HTML 3.2) - Table Row

Создает новый ряд (строку) ячеек таблицы. Ячейки в ряду создаются с помощью элементов TD и TH

Атрибуты

ALIGN - определяет способ горизонтального выравнивания содержимого всех ячеек данного ряда. Возможные значения: `left`, `center`, `right`.

VALIGN - определяет способ вертикального выравнивания содержимого всех ячеек данного ряда. Возможные значения: `top`, `bottom`, `middle`.

BGCOLOR - определяет цвет фона для всех ячеек данного ряда. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов.

TD и TH

(HTML 3.2) - Table Data & Table Head

Элемент TD создает ячейку с данными в текущей строке. Элемент TH также создает ячейку, но определяет ее как ячейку-заголовок. Такое разграничение позволяет браузерам оформлять содержимое ячейки-заголовка и ячеек с данными разными шрифтами. Кроме того, должна улучшиться работа браузеров, использующих речевой интерфейс. В качестве содержимого ячейки можно использовать другие таблицы.

Атрибуты

ALIGN - определяет способ горизонтального выравнивания содержимого ячейки. Возможные значения: `left`, `center`, `right`. По умолчанию способ выравнивания определяется значением атрибута ALIGN элемента TR. Если же и он не задан, то для TD выполняется выравнивание по левому краю, а для TH - центрирование.

VALIGN - определяет способ вертикального выравнивания содержимого ячейки. Возможные значения: `top`, `bottom`, `middle`. По умолчанию происходит выравнивание по центру (`VALIGN="middle"`), если значение этого атрибута не было задано ранее в элементе TR.

WIDTH - определяет ширину ячейки. Ширина задается в пикселах или в процентном отношении к ширине таблицы.

HEIGHT - определяет высоту ячейки. Высота задается в пикселах или в процентном

отношении к высоте таблицы.

COLSPAN - определяет количество столбцов, на которые простирается данная ячейка. По умолчанию имеет значение 1.

ROWSPAN - определяет количество рядов, на которые простирается данная ячейка. По умолчанию имеет значение 1.

NOWRAP - блокирует автоматический перенос слов в пределах текущей ячейки. Обратите внимания на примечание, касающееся использования данного атрибута (далее, внизу страницы).

BGCOLOR - определяет цвет фона ячейки. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов.

BACKGROUND - заполняет ячейку фоновым рисунком. Необходимо указать URL рисунка. Данный атрибут не работает в старых версиях браузера Netscape (до 3.X включительно).

Примеры таблиц

Примечания

- Границы ячейки отображаются только в том случае, когда она имеет некое содержание. Чтобы получить пустую ячейку с границами, достаточно поместить в нее специальный символ   или маленькую прозрачную gif-картинку.
- Если вы используете одновременно атрибуты NOWRAP и WIDTH="x", где *x* - маленькое число, то следует дополнительно вставлять внутрь ячейки <NOBR>

Пример:

```
<TABLE BORDER="1">
<CAPTION ALIGN="bottom">Заголовок таблицы</CAPTION>
<TR>
<TD>Ячейка таблицы</TD>
</TR>
</TABLE>

<!-- Так делать неправильно -->
<TABLE border=1>
    ...
    <TR><TD WIDTH=5 NOWRAP>Текст, который не должен переноситься</TD></TR>
    ...
</TABLE>
<!-- А вот так - правильно и работает всегда -->
<TABLE border=1>
...
    <TR><TD WIDTH=5 NOWRAP><NOBR>Текст, который не будет
переноситься</NOBR></TD></TR>
...
</TABLE>
```

Задание Выполните пример

Таблицы в HTML можно сделать похожими на EXEL-евские таблицы. Например мы можем объединить ячейки:

- **rowspan=n**, где n- число объединяемых ячеек по горизонтали
- **colspan=n**, где n- число объединяемых ячеек по вертикали

Данные в ячейках можно выровнять при помощи атрибута align. Этот атрибут можно применять как ко всей таблице, так и к отдельной ячейке.

Мы можем задавать определенную ширину ячейки при помощи атрибутов height (высота) и width (ширина).

Что такое CSS

CSS (*Cascading Style Sheets* — каскадные таблицы стилей) – одна из базовых технологий в современном Интернете. Нечасто можно встретить сайт, свёрстанный без примененения CSS.

CSS-код – это список инструкций для браузера, – *как и где* отображать элементы веб-страницы, написанный особым образом. Под «элементами» обычно подразумеваются теги XHTML/HTML и их содержимое.

Инструкции CSS удобно хранить в виде отдельного текстового файла с расширением *.css*, либо в виде отдельного текстового фрагмента в начале XHTML/HTML-документа (см. Включение CSS в HTML документ).

Основная идея CSS в том, чтобы отделить дизайн документа от его содержимого. CSS отвечает за оформление и внешний вид, а XHTML/HTML — за содержание и логическую структуру документа.

Посмотрим на фрагмент XHTML-документа:

```
<h1>Сказка</h1>
<p>В одной далёкой стране, на краю болота, под пеньком, жил ёжик. И вот однажды …
</p>
```

Из служебной XHTML разметки мы видим только элемент заголовка `h1` и абзаца `p`, и ни слова об оформлении — шрифтах, цвете текста, фоне, отступах и прочем дизайне. Всё это возложено на CSS:

```
/* оформляем заголовки: */
h1 {
  color: red;
  background-color: yellow;
  font: Tahoma 2em;
}
/* оформляем абзацы текста: */
p {
  color: grey;
  line-height: 150%;
}
```

Включение CSS в HTML документ

Для того, чтобы применить таблицу стилей к HTML-документу, мы можем избрать один из трёх способов, либо комбинировать их:

- применить *внешние стили* (в виде отдельного текстового *.css*-файла) с помощью элемента `link`
- *встроить стили* непосредственно в HTML-документ (в виде блока css-текста) с помощью элемента `style`
- применить *inline-стиль*, то есть назначить стиль конкретному HTML-элементу непосредственно в документе, с помощью HTML-атрибута `style`

Разберём эти способы подробнее.

Внешние стили (external style sheets)

Применяются с помощью элемента `link`, который должен располагаться внутри элемента `head` и нигде более.

<link rel=”stylesheet” type=”text/css” href=”mystyle.css” media=”all” />

Встретив в HTML-документе этот тег, браузер загрузит с сайта CSS-файл (в нашем случае это *mystyle.css*) и применит к документу содержащиеся в нём стили. Файл не должен содержать ничего, кроме CSS-инструкций.

Внешний файл со стилями удобен тем, что одни и те же стили можно применять ко множеству документов на сайте — в каждом из них достаточно лишь вписать одну строку с элементом `link`.

Таблицы стилей документа (document style sheets)

Называются так потому, что располагаются непосредственно в HTML-документе и применяются только к нему. Иногда называются *embedded style sheet* (встроенный стиль).

CSS-стили и комментарии располагаются между открывающим и закрывающим тегами элемента `style`:

<style type=”text/css”>
...
</style>

Сам тег `style` (в отличие от `link`) может находиться в любой части документа, но обычно его размещают внутри элемента `head`.

Стили, подставляемые в строку (inline styles)

Иногда нужно назначить стиль отдельному элементу на странице, не применяя внешних стилей и элемента `style`. Типичный случай — элемент встречается единожды в документе или на сайте, но требует особого оформления. Воспользуемся атрибутом `style` (именно атрибутом элементов, а не элементом!):

<p style=”color: red”>Я абзац, выделенный красным цветом, других таких на сайте нет</p>

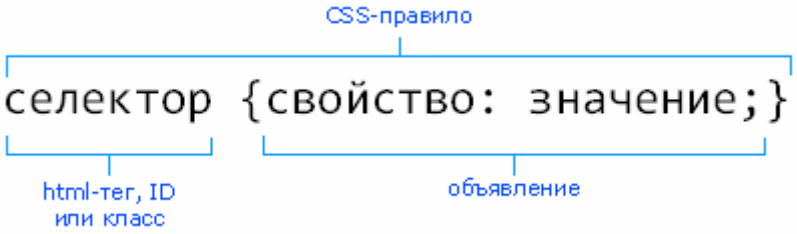
Атрибут `style` есть почти у всех HTML-элементов. Кроме тех, что располагаются вне элемента body.

Внутри атрибута `style` можно написать несколько CSS объявлений, разделённых точкой с запятой, фигурные скобки не используются.

Замечание: избегайте использования вышеописанного способа. Inline-стили смешивают содержимое документа и его дизайн, в то время как идеологически более правильно отделять содержимое документа и информацию о его оформлении.

Синтаксис CSS

Все CSS-правила состоят из *селектора* и *блока объявлений* (заключённого в фигурные скобки). Внутри блока объявлений (внутри фигурных скобок, проще говоря) может находиться одно или несколько объявлений, разделённых точкой с запятой. Объявление – это строка, составленная из *css-свойства* и его *значения*.



Как обычно выглядит css-правило

Как это может выглядеть на практике? Вот три примера CSS-правил:

```
a {text-decoration: none;}

p.announce {border: 1px dashed black;} /* здесь селектор - p.announce */

p.note {
  display: block; /* да, объявлений может быть несколько */
  float: right;   /* и их не обязательно писать на одной строке */
}
```

Каждое правило начинается с селектора (по-русски – указателя), указывающего на те html-элементы, к которым мы собираемся применить css-правило.

В блоке объявлений происходит самое интересное – мы устанавливаем правила оформления выбранных нами элементов, переопределяя их свойства – размеры, цвет, бордюры, поля, положение на экране и т.д. Основная часть этого справочника состоит именно в описании этих свойств и их возможных значений.

Селекторы

Чтобы применить css-оформление к HTML-элементу или множеству элементов, обычно используются *селекторы* – специальные указатели на HTML-объекты, к которым мы планируем применить css-правило.

Вот три основных вида селекторов.

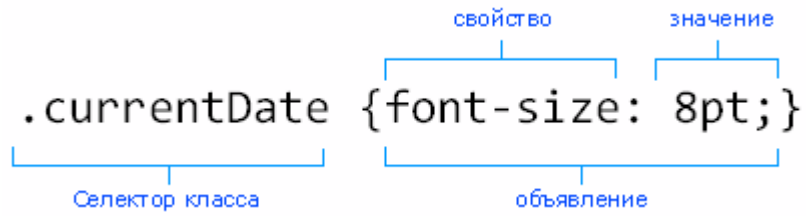
HTML селекторы

Это простейший случай – в качестве селектора мы используем имя того html-элемента, который хотим изменить. Например, для тега `<strong>` селектором будет `strong`. Соответственно, для тега `<h1>` селектором будет `h1`, и так далее. Теперь мы можем переопределить внешний вид всех таких элементов в нашем документе:

```
strong {font-weight: normal; color: red;}
h1 { font: bold 10pt verdana; }
```

Селекторы класса

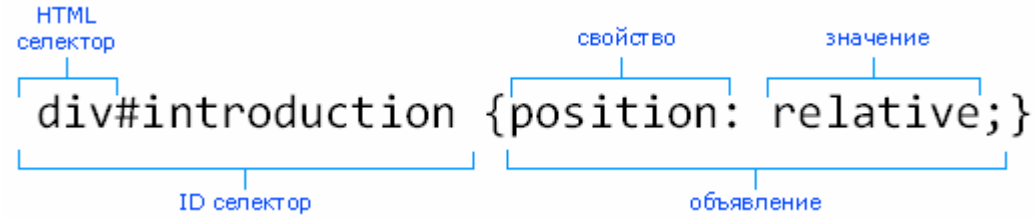
«Класс» - это некое имя, строка, которое мы можем применить к любым HTML-тегам, чтобы впоследствии ссылаться на них по имени класса. В качестве имени класса вы можете использовать практически любую строку. Удобство таких селекторов в том, что можно присвоить одно имя класса множеству html-тегов в документе и затем управлять их внешним видом, обращаясь к ним по имени класса:



.myClass { font: bold 10pt verdana; } /* меняем шрифт для всех тегов с классом myClass */

ID селекторы (или идентификаторы)

Любой идентификатор (ID) – это некое имя, которое вы, так же, как и в случае с классами, можете применить к любому HTML-тегу. Основное отличие – ID должен быть уникален в рамках html-документа:



#myObj { margin: 1em; } /* изменяем поля для элемента, у которого id=”myObj” */
span#today { margin: 1em; } /* изменяем поля для элемента span, у которого id=”today” */

Как применить один стиль к нескольким селекторам

Очень распространённая задача – применить один набор правил к нескольким разным селекторам. Это делается элементарно – достаточно перечислить селекторы через запятую:

Впоследствии оформление для селекторов можно переопределить индивидуально:

/* все параграфы и списки делаем красными, шрифтом Tahoma */
p, li, ul, ol {color:red; font-face: Tahoma, sans-serif;}
/* переопределяем цвет на синий для нумерованных списков */
ol {color: blue;}

Селекторы, зависящие от контекста

Мы научились устанавливать стили для элементов HTML независимо от того, где именно в документе они расположены. Но CSS чуть гибче, чем кажется. Мы можем «прицеливаться» точнее и определять стили для элементов в зависимости от контекста (англ. Contextual Selectors).

Вот, посмотрите:

```
/* все ссылки, находящиеся внутри списков, станут красными: */
li a {color: red;}
/* все ссылки в параграфах, находящихся внутри таблиц, станут зелеными: */
table p a {color: green;}
```

Это самый распространённый метод создания контекстуальных селекторов, который называется «селектором потомков». Разберём его подробнее.

Селекторы потомков

«Потомками» элемента HTML называются любые вложенные в него элементы: это его «дети» (то есть непосредственно вложенные), дети его детей, и так далее, вглубь иерархии тегов.

Правильно используя селекторы, мы можем прицельно применить CSS стили к нужным элементам, сославшись на их родительский элемент. Для этого перед селектором искомого элемента надо вставить селекторы его «предков», разделив их пробелом.

В приведённом примере мы вроде бы создали список селекторов, разделённый пробелами. На самом деле это *один* селектор потомков. Он указывает на все гиперссылки (элементы A) вложенные в списки (элементы LI), в свою очередь находящиеся внутри элемента с идентификатором footer. У всех таких гиперссылок мы меняем толщину шрифта на bold (полужирный шрифт).

Селекторы детей

«Детьми» или «дочерними элементами» элемента HTML называются непосредственно вложенные в него элементы, он для них является «родительским» элементом. Элементы, находящиеся на 2-м и более глубоких уровнях вложенности, «детьми» не являются – это дети детей, то есть «потомки» (см. предыдущий раздел).

CSS позволяет нам создать селектор для выбора дочерних элементов любого элемента и изменить их свойства, применив CSS стили. Для этого перед селектором искомого элемента надо вставить селекторы его «предков», разделив их знаком >.

В приведённом примере мы находим все элементы B, вложенные непосредственно в элементы I, и выключаем для них полужирный шрифт. Все остальные элементы B в документе останутся без изменений.

Селекторы смежных элементов (Adjacent Sibling Selectors)

Иногда нам надо выбрать элемент, расположенный в HTML-документе непосредственно за другим элементом. Так, например, чтобы выбрать все заголовки H1, следующие за параграфами P, и изменить их верхний отступ, мы напишем следующее правило CSS:

Важно: будет выбран только первый (!) заголовок H1, расположенный непосредственно после P. Если между элементами P и H1 встретится любой элемент, хоть один (даже если это) – то селектор не сработает и правило не применится.

Важно: селекторы смежных элементов появились в CSS v2 и поддерживаются, увы, не во всех браузерах. В Internet Explorer они появились только с 7-й версии, в Opera – с 5-й версии.

Обзор свойств CSS

Справочник опирается на актуальную сейчас спецификацию CSS 2.1, соответственно, приведенный ниже список css-свойств ограничен рамками спецификации.

Основные свойства

Список базовых свойств, которые должен знать даже начинающий веб-мастер:

`margin`, `padding`, `border`, `background-color`, `color`, `font-family`, `font-size`, `float`

Фон

`background`
: Сокращенный вариант записи для свойств `background-color`, `background-image`, `background-repeat`, `background-attachment` и `background-position`.

`background-attachment`

Устанавливает, должна ли фоновая картинка скролиться или должна быть зафиксирована в окне браузера.

`background-color`
: Устанавливает цвет фона для элемента.

`background-image`
: Устанавливает фоновую картинку для элемента.

`background-position`
: Устанавливает первоначальное положение для фоновой картинки.

`background-repeat`
: Управляет циклическим повторением фоновой картинки.

Рамка (граница, бордюр)

Влияют на все четыре рамки

`border`
: Краткий вариант записи для свойств `border-width`, `border-style` и `border-color`. Влияет на все четыре границы элемента.

`border-color`
: Устанавливает цвет рамки со всех сторон элемента.

`border-width`
: Устанавливает толщину рамки со всех сторон элемента.

`border-style`
: Определяет стиль оформления рамки вокруг элемента.

`border-collapse`
: Указывает ячейкам таблицы, иметь ли собственный бордюр или общий с соседней ячейкой.

`border-spacing`
: Устанавливает расстояние между ячейками таблицы.

Верхняя рамка

`border-top`
: Краткий вариант доступа к свойствам `border-top-width`, `border-top-style` и `border-top-color`.

`border-top-color`
: Устанавливает цвет верхнего бордюра.

`border-top-style`
: Устанавливает стиль линии верхнего бордюра.

`border-top-width`
: Устанавливает ширину верхнего бордюра.

Нижняя рамка

`border-bottom`
: Краткий вариант доступа к свойствам `border-bottom-width`, `border-bottom-style` и `border-bottom-color`.

`border-bottom-color`
: Устанавливает цвет нижнего бордюра.

`border-bottom-style`

Устанавливает стиль линии нижнего бордюра.

`border-bottom-width`
Устанавливает ширину нижнего бордюра.

Левая рамка

`border-left`
Краткий вариант доступа к свойствам `border-left-width`, `border-left-style` and `border-left-color`.

`border-left-color`
Устанавливает цвет левого бордюра.

`border-left-style`
Устанавливает стиль линии левого бордюра.

`border-left-width`
Устанавливает ширину левого бордюра.

Правая рамка

`border-right`
Краткий вариант доступа к свойствам `volume`, `border-right-style` и `border-right-color`.

`border-right-color`
Устанавливает цвет правого бордюра.

`border-right-style`
Устанавливает стиль линии правого бордюра.

`volume`
Устанавливает ширину правого бордюра.

Шрифт

`font`
Краткий вариант записи свойств `font-style`, `font-variant`, `font-weight`, `font-size`, `line-height` и `font-family`.

`font-family`
Определяет шрифт(ы) для отображения текста.

`font-size`
Управляет размером шрифта.

`font-style`
Управляет наклоном шрифта (курсив).

`font-variant`
Управляет внешним видом строчных букв (строчные как прописные, "капитель").

`font-weight`
Управляет толщиной (насыщенностью) шрифта.

Позиционирование

`position`
Определяет порядок, в соответствии с которым элемент отображается на веб-странице.

`bottom`

Сдвигает элемент относительно нижнего края. Используется совместно со свойством `position`.

`left`
: Сдвигает элемент относительно левого края. Используется совместно со свойством `position`.

`page-break-before`
: Сдвигает элемент относительно верхнего края. Используется совместно со свойством `position`.

`right`
: Сдвигает элемент относительно правого края. Используется совместно со свойством `position`.

`z-index`
: Определяет порядок, в соответствии с которым элементы накладываются друг на друга, если необходимо отобразить их на одном месте.

Форматирование

`clear`
: Запрещает/разрешает элементу обтекать "floated" объекты.

`clip`
: Определяет область элемента, которая должна отображаться на странице.

`display`
: Изменяет базовые свойства элементов.

`float`
: Сдвигает элемент к правому или левому краю.

`height`
: Определяет высоту прямоугольной области вокруг элемента.

`overflow`
: Определят как отображать блочный элемент в случае, если его содержимое выходит за рамки родительского элемента.

`visibility`
: Управляет настройкой видимости элемента.

`width`
: Определяет ширину прямоугольной области вокруг элемента.

Списки

`list-style`
: позволяет одновременно задать три параметра для маркеров пунктов списка: `list-style-type`, `list-style-position` и/или `list-style-image`.

`list-style-image`
: Устанавливает изображение, которое будет использоваться для маркировки пунктов списка.

`list-style-position`
: Определяет, как отобразить на странице маркер пункта в списке: внутри того же прямоугольника, в котором располагается элемент списка или вне его.

`list-style-type`
: Определяет, какой вид будет иметь маркер пункта в списке..

Текст

`direction`
: Применяется при создании сайтов на языках, в которых чтение страницы идет справа налево.

`letter-spacing`
: Определяет длину интервала между буквами.

`page-break-inside`
: Определяет размер межстрочного интервала.

`text-align`
: Выравнивает содержимое блочного элемента (текст или изображение) относительно границ блока, а так же содержимое ячеек таблицы по горизонтали.

`text-decoration`
: Определяет, какой оформительский прием нужно применить к тексту.

`text-indent`
: Определяет размер отступа первой строки в тексте.

`text-transform`
: Управляет написанием прописных или строчных букв в тексте.

`vertical-align`
: Определяет высоту содержимого внутри инлайн элемента или внутри ячейки таблицы.

`white-space`
: Определяет как отображать пробелы, символы табуляции и пустой строки.

`word-spacing`
: Определяет расстояние между словами.

```
[collapse collapsed]
```

Печать

`widows`
: Позволяет избежать появления висячей строки.

`orphans`
: Позволяет избежать появления одинокой первой строки.

`page-break-after`
: Определяет наличие или отсутствие разрыва страницы после элемента при печати.

`page-break-before`
: Определяет наличие или отсутствие разрыва страницы перед элементом при печати.

`page-break-inside`
: Определяет наличие или отсутствие разрыва страницы внутри элемента при печати.

Поля

`padding`
: Сокращенный способ задать следующие параметры: `padding-top`, `padding-right`, `padding-bottom` и/или `padding-left`.

`padding-bottom`
: Определяет ширину пространства между содержимым элемента и нижним бордюром.

`padding-left`
: Определяет ширину пространства между содержимым элемента и левым бордюром.

`padding-right`
: Определяет ширину внешнего пространства между правым бордюром и невидимой

границей прямоугольника.

`padding-top`
: Определяет ширину внешнего пространства между верхним бордюром и невидимой границей прямоугольника.

Прочее

`caption-side`
: Определяет, где будет отображаться заголовок таблицы: над ней или под ней.

`color`
: Устанавливает цвет текста элемента.

`content`
: Применяется для того, чтобы вставить текст или изображение до или после какого-либо элемента.

`counter-increment`
: Задает значения приращений счетчика.

`counter-reset`
: Устанавливает идентификатор, который хранит счетчик отображений какого-либо элемента и устанавливает начальное значение этого счетчика.

`cursor`
: Определяет вид курсора при наведении мышки на некий элемент.

`empty-cells`
: Определяет, нужно ли отображать границы и фон ячейки, если в ней нет содержимого.

`margin`
: Сокращенный способ задать следующие параметры: `margin-top`, `margin-right`, `margin-bottom` и/или `margin-left`

`margin-bottom`
: Определяет ширину внешнего пространства между нижним бордюром и невидимой границей прямоугольника.

`margin-left`
: Определяет ширину внешнего пространства между левым бордюром и невидимой границей прямоугольника.

`margin-right`
: Определяет ширину внешнего пространства между правым бордюром и невидимой границей прямоугольника.

`margin-top`
: Определяет ширину внешнего пространства между верхним бордюром и невидимой границей прямоугольника.

`max-height`
: Определяет максимальную высоту элемента.

`max-width`
: Определяет максимальную ширину элемента.

`min-height`
: Определяет минимальную высоту элемента.

`min-width`
: Определяет минимальную ширину элемента.

`outline`
: Это быстрый способ задать следующие параметры: `outline-width`, `outline-style` и/или `outline-color`.

`outline-color`
: Определяет цвет контура вокруг элемента.

`outline-style`
: Определяет вид контура вокруг элемента.

`outline-width`
: Определяет ширину контура вокруг элемента.

`quotes`
: Определяет вид открывающей и закрывающей кавычки в тексте.

`table-layout`
: Определяет ширину столбцов в таблице.

Задание: Примените указанные свойства к различным тегам вашего проекта, и добейтесь изменениями свойств CSS улучшения стиля вашего проекта.

Краткий обзор CSS3

Границы

Закругленные углы

```
    .border_rounded {
    background-color: #ddccb5;
    -moz-border-radius: 5px;
    -webkit-border-radius: 5px;
    border: 2px solid #897048;
    padding: 10px;
    width: 310px;
}
```

Градиентные границы

```
    .border_gradient {
    border: 8px solid #000;
    -moz-border-bottom-colors: #897048 #917953 #a18a66 #b6a488
    #c5b59b #d4c5ae #e2d6c4 #eae1d2;
    -moz-border-top-colors: #897048 #917953 #a18a66 #b6a488
    #c5b59b #d4c5ae #e2d6c4 #eae1d2;
    -moz-border-left-colors: #897048 #917953 #a18a66 #b6a488
    #c5b59b #d4c5ae #e2d6c4 #eae1d2;
    -moz-border-right-colors: #897048 #917953 #a18a66 #b6a488
    #c5b59b #d4c5ae #e2d6c4 #eae1d2;
    padding: 5px 5px 5px 15px;
    idth: 300px;
    }
```

Тени

```
.border_shadow {
-webkit-box-shadow: 10px 10px 5px #888;
padding: 5px 5px 5px 15px;
width: 300px;
}
```

Изображения на границах

```
.border_image {
-webkit-border-image: url(border.png) 27 27 27 27 round round;
}
```

Текстовые эффекты в CSS3

Тень от тектса

```
.text_shadow {
color: #897048;
background-color: #fff;
text-shadow: 2px 2px 2px #ddccb5;
font-size: 15px;
}
```

Пример текста с тенью, сделанной средствами CSS 3

Перенос длинных слов

```
.text_wrap {
word-wrap: break-word;
}
```

В этом параграфе содержаться ну простосупермегаархигипердлинны еслова, но они могутразрыватьсят акчтобынерастянуть блок

Использование своих шрифтов

```
@font-face {
```

```
font-family: 'Имя вашего шрифта';
src: url('http://vremenno.net/files/fonts/font.ttf');
}
```

Пользовательский интерфейс

Растягивание блоков

```
.ui_resizable {
padding: 20px;
border: 1px solid;
resize: both; //или inline
overflow: auto;
}
```

Этот блок можно растягивать, потянув мышкой за угол

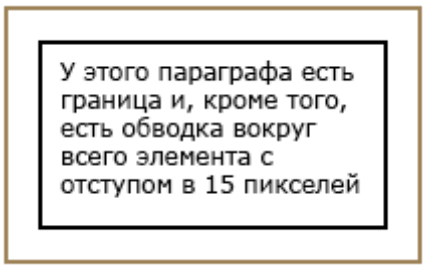
Размеры блоков

```
.area {
width: 300px;
border: 10px solid #ddccb5;
height: 60px;
}
.boxes {
box-sizing: border-box;
width:50%;
height: 60px;
text-align: center;
border: 5px solid #897048;
padding: 2px;
float:left;
}
```

Блок слева | Блок справа

Выделение

```
.ui_outline {
width: 150px;
padding: 10px;
height: 70px;
border: 2px solid black;
outline: 2px solid #897048;
outline-offset: 15px;
}
```

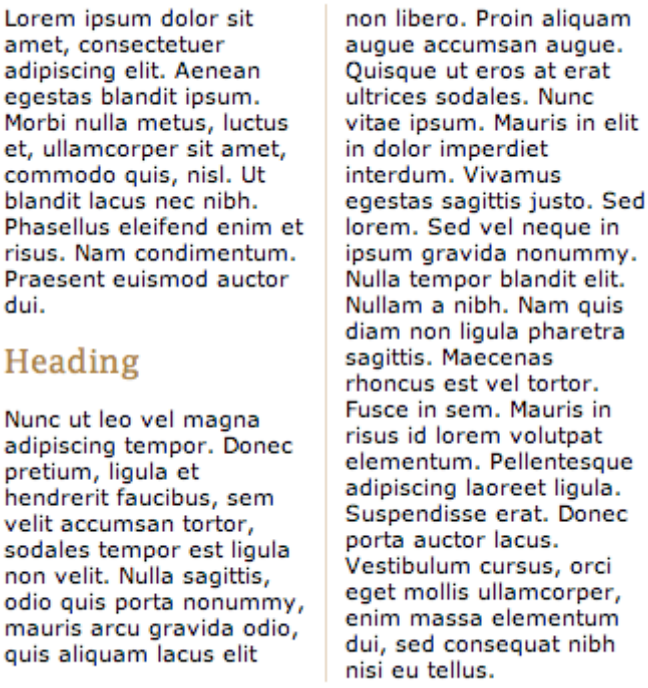


Мультиколоночность в CSS3

Вот четыре основных свойства, использующиеся для мультиколоночности:

- column-count
- column-width
- column-gap
- column-rule

```
.multiplecolumns {
-moz-column-width: 130px;
-webkit-column-width: 130px;
-moz-column-gap: 20px;
-webkit-column-gap: 20px;
-moz-column-rule: 1px solid #ddccb5;
-webkit-column-rule: 1px solid #ddccb5;
}
```



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean egestas blandit ipsum. Morbi nulla metus, luctus et, ullamcorper sit amet, commodo quis, nisl. Ut blandit lacus nec nibh. Phasellus eleifend enim et risus. Nam condimentum. Praesent euismod auctor dui.

Heading

Nunc ut leo vel magna adipiscing tempor. Donec pretium, ligula et hendrerit faucibus, sem velit accumsan tortor, sodales tempor est ligula non velit. Nulla sagittis, odio quis porta nonummy, mauris arcu gravida odio, quis aliquam lacus elit non libero. Proin aliquam augue accumsan augue. Quisque ut eros at erat ultrices sodales. Nunc vitae ipsum. Mauris in elit in dolor imperdiet interdum. Vivamus egestas sagittis justo. Sed lorem. Sed vel neque in ipsum gravida nonummy. Nulla tempor blandit elit. Nullam a nibh. Nam quis diam non ligula pharetra sagittis. Maecenas rhoncus est vel tortor. Fusce in sem. Mauris in risus id lorem volutpat elementum. Pellentesque adipiscing laoreet ligula. Suspendisse erat. Donec porta auctor lacus. Vestibulum cursus, orci eget mollis ullamcorper, enim massa elementum dui, sed consequat nibh nisi eu tellus.

Объединение колонок

```
h2 {
column-span: all;
}
```

Фоновые изображения

Размер фоновой картинки

```
.backgroundsize {
background: url(/files/images/logo.gif);
-webkit-background-size: 203px 45px;
-khtml-background-size: 203px 45px;
-o-background-size: 203px 45px;
background-size: 203px 45px;
background-repeat: no-repeat;
padding: 60px 5px 5px 40px;
border: 3px solid #ddccb5;
}
```

Комбинирование фоновых изображений

```
.multiplebackgrounds {
height: 150px;
width: 270px;
padding: 40px 20px 20px 20px;
background: url(top.gif) top left no-repeat, url(bottom.gif) bottom left no-
repeat, url(middle.gif) left repeat-y;
}
```

Начало отчета координат фона

CSS3 позволяет вам указывать как вычислять положение фоновой картинки — от обводки (border), внутреннего отступа (padding) или содержания (content).

Свойство background-clip

Позволяет наложить любое изображение на текст, границы или тень, реализую при этом что-то похожее на маску в Photoshop.

```
.background_clip {
background: url(green-background.png), black;
border: 5px solid black;
padding: 5px;
-webkit-background-clip: text, border;
color: transparent;
}
```

Селекторы

Выбор по свойствам элемента

```
[att^=val]элемент, значение свойства (att) которого начинается с 'val'.
[att$=val]
значение свойства (att) которого заканчивается на 'val'.
[att*=val]
элемент, значение свойства (att) которого содержит по крайней мере одно
вхождение подстроки 'val'.
```

Терминология

- **Элемент**(element)- конструкция языка HTML. Это контейнер, содержащий данные и позволяющий отформатировать их определенным образом. Любая Web-страница представляет собой ряд элементов. Одна из идей гипертекста- возможность вложения различных элементов.
- **Тег**(tag)- начальный или конечный маркеры элемента. Теги определяют границы действия элементов и отделяют элементы друг от друга. В тексте Web- страницы теги

заключаются в угловые скобки, а конечный тег ещё с косой чертой (/)

- **Атрибут**(atrubute)- параметр или свойства элемента. Это, по сути, переменная, которой может придаваться определенное значение в кавычках. Атрибуты расположены внутри начального тега и отделяются они пробелами.
- **Гиперссылка**(hyperlink)- фрагмент гипертекста, который указывает на другой файл или объект.
- **Фрейм**(frame)- этот термин имеет два значения. Первое- область документа со своими полосами прокрутки. Второе- один кадр сложного анимированного изображения.
- **HTML-файл** или HTML-страница- документ, созданный в виде гипертекста на основе языка HTML.
- **Апплет**(applet) -программа, передаваемая на компьютер клиента в виде отдельно файла и запускаемая при просмотре Web- страниц.
- **Скрипт** или Сценарий(script) -программа лежащая в HTML- страницы, расширяя её возможность при помощи средств мультимедиа.
- **Расширение** (extension) — элемент, не входящий в спецификацию языка, но использующийся, обеспечивая возможность создания нового интересного эффекта форматирования.
- **CGI** (Common Gateway Interface) — общее название для программ, которые, работая на сервере, позволяют расширить возможности Web-страниц. Например, без таких программ невозможно создание интерактивных страниц. К таким программам относят виртуальные магазины, некоторые чаты и т.д.
- **Программный код** или просто код — аналог понятия «текст программы».
- **Код HTML** — гипертекстовый документ в своем первоначальном виде, когда видны все элементы и атрибуты.
- **World Wide Web, WWW** или просто Web — Всемирная паутина, распределенная система доступа к гипертекстовым документам, существующая в Интернете. HTML является основным языком для создания документов в WWW. Изучая его, мы, фактически, изучаем часть этой системы, хотя область применения языка намного шире.
- **Web-страница** — документ (файл), подготовленный в формате гипертекста и размещенный в World Wide Web.
- **Сайт** (site) — набор Web-страниц, принадлежащих одному владельцу.
- **Броузер** (browser) — программа для просмотра Web-страниц.
- **Загрузка** (downloading) — копирование файлов с сервера на компьютер-клиент.
- **URL** (Uniform Resource Locator) или универсальный указатель ресурса — адрес некоторого объекта в Интернете. Типичный URL для WWW имеет вид: http://www.название.домен/имя файла Здесь название — это часть адреса, который часто употребляется для обозначения владельца сайта, а домен — обозначение крупного «раздела» Интернета: страны, области деятельности и т. д. URL используются для того, чтобы указать конкретную Web-страницу или графический файл в гиперссылках, а также везде, где требуется однозначно определить месторасположение Web-страницы или файла.

- **Базовый URL** — часть адреса, которая является общей для всех ссылок текущей Web-страницы.
- **Базовый цвет** — каждый цветовой оттенок на экране монитора получается соединением трех базовых цветов: красного, зеленого и синего.
- **Цветовой канал** - интенсивность красного, синего, зеленого цветов. Цвет пикселей определяется этой величиной.