КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра технологий программирования

А.И. ЕНИКЕЕВ

Э.Р. СТЕПАНОВА

СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ. ОСНОВЫ WEB-ПРОГРАММИРОВАНИЯ

Учебно-методическое пособие

Казань – 2015

УДК 821.111.09
ББК Ш3(4)

Принято на заседании кафедры технологий программирования
Протокол № 4 от 23 апреля 2015 года

Еникеев А.И., Степанова Э.Р.
Современные информационные технологии.
Основы web-программирования/ А.И. Еникеев,
Э.Р. Степанова – Казань: Казан. ун-т, 2015. – 80 с.

Информационная технология является более емким понятием, отражающим современное представление о процессах преобразования информации информационном обществе. В умелом сочетании двух информационных технологий — управленческой и компьютерной — залог успешной работы информационной системы.

Одним из главных составляющих современного программирования, является web-программирование. В данном учебно-методическом пособии представлены основы html – языка разметки гипертекста и использование css-стилей.

Оглавление

ВВЕДЕНИЕ

Информационная технология является наиболее важной составляющей процесса использования информационных ресурсов общества. К настоящему времени она прошла несколько эволюционных этапов, смена которых определялась главным образом развитием научно-технического прогресса, появлением новых технических средств переработки информации. В современном обществе основным техническим средством технологии переработки информации служит персональный компьютер, который существенно повлиял как на концепцию построения и использования технологических процессов, так и на качество результатной информации. Внедрение персонального компьютера в информационную сферу и применение телекоммуникационных средств связи определили новый этап развития информационной технологии и, как следствие, изменение ее названия за счет присоединения одного из синонимов: "новая", "компьютерная" или "современная".

Прилагательное "новая" подчеркивает новаторский, а не эволюционный характер этой технологии. Ее внедрение является новаторским актом в том смысле, что она существенно изменяет содержание различных видов деятельности в организациях. В понятие новой информационной технологии включены также коммуникационные технологии, которые обеспечивают передачу информации разными средствами, а именно — телефон, телеграф, телекоммуникации, факс и др. В табл. 1.1 приведены основные характерные черты новой информационной технологии.

Таблица 1.1

Основные характеристики новой информационной технологии

Методология	Основной признак	Результат
Принципиально новые средства обработки информации	"Встраивание" в технологию управления	Новая технология коммуникаций
Целостные технологические системы	Интеграция функций специалистов и менеджеров	Новая технология обработки информации
Целенаправленные создание, передача, хранение и отображение информации	Учет закономерностей социальной среды	Новая технология принятия управленческих решений

Новая информационная технология — информационная технология с "дружественным" интерфейсом работы пользователя, использующая персональные компьютеры и телекоммуникационные средства.

Основным техническим средством реализации НИТ является компьютер. В основе реализации НИТ лежат три основных принципа:

· интерактивный (диалоговый) режим работы с компьютером;

· интегрированность (стыковка, взаимосвязь) с другими программными продуктами;

· гибкость процесса изменения как данных, так и постановок задач.

Термин новая отражает в ее структуре не только технологии, основанные на использовании компьютеров, но и технологии, основанные на других технических средствах, особенно на средствах, обеспечивающих телекоммуникацию.

Информационная технология является более емким понятием, отражающим современное представление о процессах преобразования информации в информационном обществе. В умелом сочетании двух информационных технологий — управленческой и компьютерной — залог успешной работы информационной системы.

Обобщая все вышесказанное, предлагаем несколько более узкие, нежели введенные ранее, определения информационной системы и технологии, реализованных средствами компьютерной техники.

Информационная технология — совокупность четко определенных целенаправленных действий персонала по переработке информации на компьютере.

Информационная система — человеко-компьютерная система для поддержки принятия решений и производства информационных продуктов, использующая компьютерную информационную технологию.

История развития технологий WWW

Всеми́рная паути́на (англ. *World Wide Web*) — распределённая система, предоставляющая доступ к связанным между собой документам, расположенным на различных компьютерах, подключенных к Интернету. Для обозначения Всемирной паутины также используют слово веб (англ. *web* «паутина») и аббревиатуру WWW.

Всемирную паутину образуют сотни миллионов веб-серверов. Большинство ресурсов всемирной паутины основаны на технологии гипертекста. Гипертекстовые документы, размещаемые во Всемирной паутине, называются веб-страницами. Несколько веб-страниц,

объединённых общей темой, дизайном, а также связанных между собой ссылками и обычно находящихся на одном и том же веб-сервере, называются веб-сайтом. Для загрузки и просмотра веб-страниц используются специальные программы — браузеры (англ. *browser*).

Изобретателями всемирной паутины считаются Тим Бернерс-Ли и, в меньшей степени, Роберт Кайо. Тим Бернерс-Ли является автором технологий HTTP, URI/URL и HTML. В 1980 году он работал в Европейском совете по ядерным исследованиям (фр. *Conseil Européen pour la Recherche Nucléaire, CERN*) консультантом по программному обеспечению. Именно там, в Женеве (Швейцария), он для собственных нужд написал программу «Энквайр» (англ. *Enquire*, можно вольно перевести как «Дознаватель») В 1989 году, работая в CERN над внутренней сетью организации, Тим Бернерс-Ли предложил глобальный гипертекстовый проект, теперь известный как Всемирная паутина. Проект подразумевал публикацию гипертекстовых документов, связанных между собой гиперссылками, что облегчило бы поиск и консолидацию информации для учёных CERN. Для осуществления проекта Тимом Бернерсом-Ли (совместно с его помощниками) были изобретены идентификаторы URI, протокол HTTP и язык HTML.В период с 1991 по 1993 год Бернерс-Ли усовершенствовал технические спецификации этих стандартов и опубликовал их. Но, всё же, официально годом рождения Всемирной паутины нужно считать 1989 год.

Для улучшения визуального восприятия веба стала широко применяться технология CSS, которая позволяет задавать единые стили оформления для множества веб-страниц. Ещё одно нововведение, на которое стоит обратить внимание, — система обозначения ресурсов URN (англ. *Uniform Resource Name*).

Популярная концепция развития Всемирной паутины — создание семантической паутины. Семантическая паутина — это надстройка над существующей Всемирной паутиной, которая призвана сделать размещённую в сети информацию более понятной для компьютеров.

Семантическая паутина — это концепция сети, в которой каждый ресурс на человеческом языке был бы снабжён описанием, понятным компьютеру. Семантическая паутина открывает доступ к чётко структурированной информации для любых приложений, независимо от платформы и независимо от языков программирования. Программы смогут сами находить нужные ресурсы, обрабатывать информацию, классифицировать данные, выявлять логические связи, делать выводы и даже принимать решения на основе этих выводов.

Введение в WEB-технологии. Основы HTML

HTML представляет собой язык разметки гипертекста, который предназначен для создания веб-страниц. Когда такая страница открывается в браузере, он просматривает код HTML, находит специальные символы, называемые тегами, и использует их для создания элементов, таких как: рисунки, таблицы, ссылки и др. Прежде чем создавать в корне сайта папки и файлы, необходимо разработать структуру сайта — какие разделы и подразделы будут присутствовать, как они будут называться. Для эффективной работы не обойтись без необходимых и привычных инструментов, в том числе и при написании кода HTML. Поэтому для начальной разработки веб-страниц или даже небольшого сайта — так называется набор страниц, связанных между собой ссылками и единым оформлением, нам понадобятся следующие программы.

- Текстовый редактор.
- Браузер для просмотра результатов.
- Валидатор — программа для проверки синтаксиса HTML и выявления ошибок в коде.
- Графический редактор.
- Справочник по тегам HTML.

Далее рассмотрим эти инструменты подробнее.

Текстовый редактор

HTML-документ можно создавать в любом текстовом редакторе, хоть Блокноте, тем не менее, для этой цели подойдет не всякая программа. Нужна такая, чтобы поддерживала следующие возможности:

- подсветка синтаксиса — выделение тегов, текста, ключевых слов и параметров разными цветами. Это облегчает поиск нужного элемента, ускоряет работу разработчика и снижает возникновение ошибок;
- работа с вкладками. Сайт представляет собой набор файлов, которые приходится править по отдельности, для чего нужен редактор, умеющий одновременно работать сразу с несколькими документами. При этом файлы удобно открывать в отдельных вкладках, чтобы быстро переходить к нужному документу;
- проверка текущего документа на ошибки.

Каждый раздел представляет собой один HTML-документ, который следует создать и дать ему имя. Имена файлов лучше называть латинскими символами без пробелов в нижнем регистре. Такой подход гарантирует универсальность и работоспособность на разных платформах.

Обязательные имена

index.html — название главной страницы, а также веб-страниц, размещаемых в папках, которые должны открываться при их указании в адресе. Это имя, как уже упоминалось, может меняться в зависимости от типа веб-сервера и его настроек. Но обычно оно именно такое.

.htaccess — конфигурационный файл веб-сервера Apache. Указанный сервер является наиболее популярным и распространенным в мире, поэтому и данный файл можно встретить повсеместно. Бывают, конечно, исключения.

robots.txt — файл, предназначенный для поисковых систем. При индексировании сайта, в первую очередь ищется он.

Как создать свой первый документ:

Откройте блокнот

Пуск> Стандартные> Блокнот и напишите в нем следующий текст:

<html>

<head>

<title> Моя первая страничка </title>

</head>

<body>

Привет мир!!!

Меня зовут (здесь впишите Ваше имя), это моя первая страничка!

</body>

</html>

Далее, сохраните этот текст как html документ, название придумайте сами. главное чтобы расширение было html

В блокноте кликаем по меню "Файл", выбираем "Сохранить как." в строке "Имя файла" пишем: index.html, просто по умолчанию блокнот предлагает сохранить файл с расширением *.txt, а нам нужно расширение *. html

Далее открываем этот файл при помощи Вашего браузера. ну к примеру того же Internet Explorera (правой кнопкой по нашему файлу. "Открыть с помощью" Internet Explorer)

То что написано между <…> - называют тегами они не видны читателю, заглянувшему на Вашу страницу, зато хорошо видны браузеру, который наткнувшись на тег <html> понимает его как сигнал к тому, что далее будет документ, который необходимо прочитать и вывести на монитор в нужном виде а вот тег </html> говорит о том что документ закончился и от браузера, больше ничего не требуется.

<html> - *начало документа*

<head> - *здесь указывается основная служебная информация о документе*

<title> - *"название" значит.. это в шапке окна нужно написать его название:*

Моя первая страничка

</title> - *все название закончилось*

</head>

<body> - *"тело" документа всё что написано ниже выставляем на всеобщее обозрение*

Привет мир!!!

 - *переносим текст на следующую строчку*

Меня зовут (здесь Ваше имя), это моя первая страничка!

</body> - *Больше ничего не отображать*

</html> - *конец*

Вот так примерно и происходит чтение нашей странички. Как видите браузер довольно своенравный тип, поэтому команды ему нужно подавать чёткие и ясные

1) Необходимо запомнить, что если есть открывающий тег <…> то обязательно должен быть и закрывающий </…>

Хотя есть и исключения, как например у нас тег
 - он закрытия не требует потому что говорит лишь о том, что следует писать с новой строки.

2) Все документы должны иметь вот такой шаблон кода:

<html>- *начало документа*

<head>- *начало головы*

</head>- *закрытие головы*

<body>- *начало тела*

</body>- *закрытие тела*

</html>- *конец документа*

Данные теги являются обязательными. Писать их необходимо всегда для каждой новой странички, и только в таком порядке.

3) О порядке:

Открывающий и закрывающий тег по типу <…> </…> представляет собой своего рода ёмкость, ящик в который могут складываться другие теги - ящички поменьше, следовательно, согласно логики документ должен выглядеть так:

<Тег "большой ящик">
<Тег "ящик средний">
<Тег "ящик маленький">
содержание
</Тег "ящик маленький" >
</Тег "ящик средний" >
</Тег "большой ящик">

Если писать, например, так:

<Тег "большой ящик">
<Тег "ящик средний">
<Тег "ящик маленький">
содержание
</Тег "большой ящик">
</Тег "ящик маленький">
</Тег "ящик средний">

Ничего не получится

Чётко структурируйте код Вашей странички иначе ничего работать не будет.

Параграф.

Параграф. это такой отрезок текста, одно или несколько предложений, который в книгах обычно печатается с новой строки, тем самым, выделяя этот текст из основной массы. Книгу, разбитую на параграфы легко читать, потому что, как правило, одному параграфу соответствует одна мысль или логическая часть текста.

Так вот, для того чтобы на странице сайта разбить текст на параграфы, необходимо воспользоваться тегом - собственно параграф.

Параграф имеет атрибут align "выравнивание" который в свою очередь может быть равен тому ли иному значению.

Рассмотрим на примерах:

С помощью параграфа можно расположить наш текст по центру:

Привет мир!!!

По левому краю:

Привет мир!!!

По правому краю:

Привет мир!!!

Или же обоим краям документа:

Привет мир!!! - здесь нужен текст подлиней чтобы эффект был хорошо виден при открытии документа

Давайте слегка изменим нашу первую страничку:

```
<html>
<head>
<title>Мой первый сайт </title>
</head>
<body>
<p align="center">Привет мир!!!</p>
<br>
<p align="justify">
Меня зовут Иван! Я живу и учусь в Казани.
</p>
</body>
</html>
```

Запомним некоторые вещи:

Тег не может содержать в себе других параграфов, то есть писать вот так:

```
<p>
<p>
</p>
</p>
```

Нельзя! - это нелогично, как может один параграф содержать в себе другой?

2) Так же, нельзя писать пустые теги без текста или других тегов.

 здесь, что-то обязательно должно быть!!!

3) По умолчанию Ваш текст выравнивается браузером по левому краю, так что если Вам так и надо атрибут align="left" для параграфа можно не указывать.

4) Тег подразумевает в себе перенос строки, если это Вам не нужно, используйте вместо тега тег данный тег является альтернативой тегу пишется так:

```
<div align="center">Привет мир!!!</div>
<div align="left">Привет мир!!!</div>
<div align="right">Привет мир!!!</div>
<div align="justify">Привет мир!!!</div>
```

Все то же самое, только данный тег не будет переносить текст на следующую строку и в него можно вставлять тег по принципу:

```
<div>
<p align="left">Пишем слева</p>
<p align="right">Пишем справа</p>
</div>
```

А вообще тег многофункциональный и по своей сути он является пустым блоком-контейнером, который может содержать в себе как текст, так и другие теги. Работа с текстом это далеко не основная задача тега , но об остальных возможностях данного тега мы поговорим позже.

5) Еще одним способом выравнивания текста по центру является использование тега любое содержание взятое в данный тег выравнивается по центру экрана. Пишется так:

Привет мир!!!

Заголовки

В наборе тегов html языка имеется шесть типов заголовков:

<h1> Привет мир!!! </h1>

<h2> Привет мир!!! </h2>

<h3> Привет мир!!! </h3>

<h4> Привет мир!!! </h4>

<h5> Привет мир!!! </h5>

<h6> Привет мир!!! </h6>

<hx> тег заголовка, где значение x является величиной буковок (может быть от одного до шести). Помните, что после использования того или иного заголовка происходит перенос строки - на то он и заголовок.

Заголовки дело хорошее и достаточно удобное, но ими можно выделять только маленькие кусочки текста, а что если нам надо выделить весь текст?

Тег в переводе на русский - "шрифт".

Тег помимо прочих атрибутов, о которых ещё пойдет речь, имеет атрибут size - размер.

Пишется и выглядит это так:

Привет мир!!!

Привет мир!!!

Привет мир!!!

Привет мир!!!

Привет мир!!!

Добавим эти теги на нашу страницу.

```
<html>
<head>
<title> Моя первый сайт </title>
</head>
<body>
<center><h2>Привет мир!!!</h2></center>
<br>
<p align=”justify”>
<font  size=”+1”>Меня зовут Иван. Живу и учусь в  <font size=”+2”>Казани </font> Чудесный город </font>
</p>
</body>
</html>
```

Цвет

Для придания страничке красивого вида не обойтись без палитры с красками.

В html языке используется своя палитра красок. Вот основные цвета, выглядят они так:

#000000 black	#fffff f white	#ff0000 red
#ffa500 orange	#ffff00 yellow	#008000 green
#00ffff cyan	#0000ff blue	#800080 purple

Один и тот же цвет можно задать двумя способами: используя шестнадцатеричное значение цвета RGB - например #008000 - зелёный цвет, либо используя константы цвета - green

 у него есть еще один атрибут - color.

Так вот, если к примеру написать так:

Привет мир!!! - То цвет шрифта станет красным.

А можно так:

Привет мир!!! - Будет тоже самое.

Но советую писать всё же шестнадцатеричным числом, во-первых, по понятным причинам не для всех оттенков цветов есть своё название, а во-вторых, поговаривают, что не все браузеры знают названия тех или иных красок.

Есть еще один способ изменить цвет текста. Тег <body></body> "тело" - имеет атрибут text - "текст" если присудить этому атрибуту любой цвет из доступной палитры то весь текст в нашей странице окрасится, кроме тех мест, где мы "принудительно" указали другой цвет.

В строке где стоит открывающий тег <body> пишем так:

<body text="#ff8c40 ">

Теперь весь текст у нас стал оранжевым кроме заголовка "Привет мир!!!" который мы отдельно перекрасили в красный.

А вот атрибут тега <body> bgcolor и его значение задает цвет фона страницы

<body bgcolor="#40caff"> - залили всё голубым.

На данном этапе получиили:

<html>

<head>

<title> Моя первый сайт </title>

</head>

<body text=”#484800” bgcolor=”#e8e8e8”>

```
<center>
<h2>
<font color=”#008000”>Привет мир!!! </font>
</h2>
</center>
<p align=”justify”>
<font  size=”+1”>Меня зовут Карлсон! Я в меру упитанный мужчина -  это моя первая страничка! Здесь я хочу найти себе новых друзей,  для того чтобы вместе гулять по  крышам! Я очень очень сильно люблю <font  size=”+2” color=”#ff0000”>варенье!!! </font> С  нетерпением буду ждать Вашего приглашения на чай. Прилечу!!
</font>
</p>
</body>
</html>
```

Обратите внимание на то, как пишется код, если в теге присутствует два и более атрибута. В нашей строчке <body text="#ff207b" bgcolor="#1a77f0"> у тега <body> два атрибута text и bgcolor мы просто пишем их подряд через пробел, не разделяя никакими другими знаками.

Стиль текста

Здесь все достаточно просто.

Итак, новые теги:

  - **Полужирный текст**

  - *Наклонный текст*

<u> </u> - <u>Подчеркнутый текст</u>

<strike> </strike> - ~~Перечеркнутый~~

<s> </s> - ~~Перечеркнутый~~ (второй вариант, от первого ничем не отличается)

<tt> </tt> - моноширинный шрифт

<small> - Малый </small>

<big> - Большой </big>

^{- Верхний индекс}

_{- Нижний индекс}

Текст заключённый между этими открывающими и закрывающими тегами приобретёт нужный нам стиль.

Шрифт

Для того чтобы изменить шрифт документа необходимо дать указание браузеру, что показывать текст таким шрифтом. Для этого используем всё тот же тег и его атрибут face .

Пишется так:

Эта строчка будет написана с помощью шрифта Arial

Пример:

<html>

<head>

<title> Использование различных шрифтов </title>

</head>

<body>

Мало кто знает, как много надо знать для того, что бы знать, как мало мы знаем.

Мало кто знает, как много надо знать для того, что бы знать, как мало мы знаем.

Мало кто знает, как много надо знать для того, что бы знать, как мало мы знаем.

</body>

</html>

Здесь необходимо отметить, что браузер использует библиотеку шрифтов, установленную на компьютере пользователя, и если вдруг указанного Вами шрифта в этой библиотеке не окажется, то он заменит его на тот который присутствует.

Предварительно отформатированный текст.

Если Вы обратили внимание, а если не обратили то знайте, что в браузерах текст набранный с помощью текстовых редакторов проходит "обработку" перед выводом его на экран компьютера. Так в набранном Вами тексте удаляются все переносы строк и лишние пробелы, пробелов между словами или просто символами может быть не более одного.

Проводится данная "обработка", для того чтобы на мониторах с разным расширением экрана текст переносился на следующую строку в тех местах где это действительно необходимо, а не там где были раннее расставлены пробелы и переносы строк.

Однако такая политика браузеров, в ряде случаев, не всегда оправданна. Как например, написать стихи? Нет можно конечно после каждой строчки писать тег
, но есть вариант куда проще.

Знакомимся тег <pre>, текст заключённый в данный тег выводится браузерами на экран в том виде в котором он был набран, т.е. со всеми пробелами и переносами строк

Пример:

```
<html>
<head>
<title>Пробелы и перенос строки</title>
</head>
<body>
<pre>


СЛОН.
Дали туфельки слону.
Взял он туфельку одну
И сказал: - Нужны пошире,
И не две, а все четыре!
С. Я. Маршак.
</pre>
</body>
</html>
```

Рисунки.

В этой главе вы узнаете, как добавить на нашу страничку графические изображения фотографию например, или рисунок, а также о том, что можно с ними вытворять используя предложенный набор тегов html языка.

Путь к файлу

Вам нужно добавить рисунок на страницу своего сайта.

Есть у Вас фотография, которая где-то лежит на Вашем жёстком диске, копируем ёе и вставляем в ту папку (директорию) где уже лежит заготовка будущей страницы, ну то есть туда же, куда сохраняете блокнотом html документ. Так, вот для того чтобы теперь вставить её в нашу страничку к ней нужно указать путь, делается это так:

```
<img src="foto.jpg">
```

Где foto.jpg это название Вашей фотографии в данном случае, так как фото лежит рядом с html документом, путь к ней мы не указываем.

Да, помните, тег <img> не требует закрывающего тега!

Ряд примеров где путь указывается:

<img src="myfoto/foto.jpg"> - Такая запись подразумевает, что в директории где расположен наш html документ есть папка myfoto в которой расположен файл foto.jpg

<img src="myfoto/graphics/foto.jpg"> - Значит рядом с документом расположена папка myfoto, в ней еще одна папка с названием graphics, и уже в ней нужная нам фотография foto.jpg которую нужно выложить для всеобщего обозрения.

<img src="./foto.jpg"> - А это значит, что фото размещено на уровень выше от документа

<img src="././foto.jpg"> - Так соответственно на два уровня выше, сколько поставите./ настолько и поднимитесь.

Так же можно указывая место фотографии, ссылаться на тот или иной интернет ресурс

<img src="http://www.site.ru/foto/foto.jpg">

Ну что, давайте попробуем выложить фото.

Вот мой пример:

```
<html>
<head>
<title>Моя первая страничка с фото</title>
</head>
<body text="#484800" bgcolor="#e8e8e8">
<center>
<h2> <font color="#008000"> Привет мир!!!</font> </h2>
</center>
<p align="justify">
<font size="+1">
```

<img src="graphics/foto.jpg">
Меня зовут Карлсон! Я в меру упитанный мужчина - это моя первая страничка! Здесь я хочу найти себе новых друзей, для того чтобы вместе гулять по крышам! Я очень очень сильно люблю вареньё!!! С нетерпеньем буду ждать Вашего приглашения на чай. Прилечу!!

</p>
</body>
</html>

Атрибуты тега <img>

Поговорим о расположении изображений относительно текста. Как и другие теги <img> тоже имеет свои атрибуты. уже знакомый нам атрибут align "выравнивание" применим и к данному тегу

<img src="foto.jpg" align="left"> - фото слева от текста
<img src="foto.jpg" align="right"> - фото справа от текста
<img src="foto.jpg" align="top"> - текст выше фото
<img src="foto.jpg" align="bottom"> - текст ниже фото
<img src="foto.jpg" align="middle"> - ну и соответственно текст посередине

Помимо align тег <img> имеет еще ряд атрибутов, но сначала узнаем немного о пикселях. Пиксель -это элементарная неделимая единица изображения. Каждый пиксель имеет свои координаты, например, самый верхний левый пиксель на мониторе имеет координаты x=1, y=1, а самый нижний правый в зависимости от разрешения монитора. x=800, y=600 - будет соответственно при разрешении 800 на 600 точек. Впрочем, эта информация пригодится нам потом, а сейчас нужно усвоить, что все расстояния в графических изображениях меряются пикселями. так картинка длинной 800 пикселей и шириной в 600 пикселей при указанном разрешении заполнит весь экран.

Теперь продолжим.

<img src="foto.jpg" vspace="15"> - Атрибут vspace задаёт расстояние по вертикали от рисунка до текста, в данном случае мы задали расстояние в 15 пикселей

<img src="foto.jpg" hspace="25"> - Расстояние по горизонтали соответственно

<img src="foto.jpg" width="180"> - Ширина непосредственно самого изображения

<img src="foto.jpg" height="240"> - Высота изображения. Если атрибуты width и height не использовать, то ширина и высота изображения по умолчанию будет равна реальным её размерам, без каких либо искажений.

<img src="foto.jpg" border="5">- Бордюр, рамка вокруг изображения и её толщина в пикселях.

<img src="foto.jpg" border="5" bordercolor="#008000 "> - bordercolor - это цвет рамки.

<img src="foto.jpg" alt="Это моя фотография!!!"> -Атрибут alt - это описание изображения. Если навести курсор на наше фото и подержать его там несколько секунд, выскочит надпись -Это моя фотография!!!

<img src="foto.jpg" title="Это моя фотография!!!"> - альтернатива alt в данном случае.

А еще изображение можно сделать фоном страницы. для этого используем атрибут background "фон" открывающего тега <body>

Вот так:

<body background="foto.jpg">

Таблицы

Помимо прочих объектов в свой сайт Вы можете вставить таблицы и сразу забегая вперёд скажу о том что они имеют не малую значимость при создании сайта. С помощью таблицы можно не только выложить ту или иную информацию, тарифную сетку или график дежурств к примеру, но и, взяв, ёе за основу полностью построить на ней свой сайт, таблицы порой незаменимы при верстке страницы, но об этом позже. А сейчас давайте научимся её рисовать.

Рисуем таблицу

Тег <table> задаёт начало и конец таблицы, любая таблица, как известно, состоит из строк и столбцов, для этого есть ещё два тега:

<tr> - строка таблицы

<td> - столбец таблицы

Вместе эти теги пишутся так:

```
<table>
<tr>
<td>ячейка</td>
</tr>
</table>
```

Такая запись это самая маленькая таблица, в ней всего одна строка, содержащая один столбец - ячейку

Поставим перед собой задачу нарисовать таблицу из трёх строк и трёх столбцов, а заодно вспомним атрибут border "рамка", который добавит нам наглядности.

```
<html>
<head>
<title>Таблица</title>
</head>
<body>
<table border="1">
<tr>
<td>строка1 ячейка1</td>
<td>строка1 ячейка2</td>
<td>строка1 ячейка3</td>
</tr>
<tr>
<td>строка2 ячейка1</td>
<td>строка2 ячейка2</td>
```

```
<td>строка2 ячейка3</td>
</tr>
<tr>
<td>строка3 ячейка1</td>
<td>строка3 ячейка2</td>
<td>строка3 ячейка3</td>
</tr>
</table>
</body>
</html>
```

Объединение ячеек.

Часто при работе с таблицами возникает необходимость объединить те или иные ячейки в одну.

Эту задачу решают атрибуты colspan и rowspan

- colspan - определяет какое количество столбцов будет занимать данная ячейка
- rowspan - количество рядов занимаемое ячейкой

Предположим что в нашем примере нам необходимо "объединить" в третьей строке вторую и третью ячейку, для этого атрибуту colspan присваиваем значение 2 (растянуть на два столбца) и вставляем его в нужное место.

```
<html>
<head>
<title>Таблица</title>
</head>
<body>
<table border="1">
<tr>
<td>строка1 ячейка1</td>
<td>строка1 ячейка2</td>
```

```
<td>строка1 ячейка3</td>
</tr>
<tr>
<td>строка2 ячейка1</td>
<td>строка2 ячейка2</td>
<td>строка2 ячейка3</td>
</tr>
<tr>
<td>строка3 ячейка1</td>
<td colspan="2">строка3 ячейка2</td>
</tr>
</table>
</body>
</html>
```

Размеры таблицы.

Если Вы самостоятельно тренировались с рисованием таблицы то наверняка обратили внимание на то, что размеры таблицы и ячеек по умолчанию ограничены вставленным в неё текстом. и "ползают" себе как хотят. Вспомните про атрибуты width - ширина и height - высота, которые мы использовали для растягивания рисунков, они так же применимы к тегам <table>, <tr>и <td>. Приведем пример. В нём заданы размеры таблицы и отдельных её ячеек, а заодно и вся таблица выровнена по центру знакомым тегом

```
<html>
<head>
<title>Таблица</title>
</head>
<body>
<center>
<table border="1" width="640" height="480">
```

```
<tr>
<td rowspan="3" width="150">строка1 ячейка1</td>
<td height="30">строка1 ячейка2</td>
<td>строка1 ячейка3</td>
</tr>
<tr>
<td height="30">строка2 ячейка2</td>
<td>строка2 ячейка3</td>
</tr>
<tr>
<td colspan="2" >строка3 ячейка2</td>
</tr>
</table>
</center>
</body>
</html>
```

Верстка страницы с использованием таблицы

```
<html>
<head>
<title>Верстка страницы</title>
</head>
<body>
<center>
<table border="1" width="640" height="480">
<tr>
<td colspan="5" height="30"><center>Заголовок</center></td>
</tr>
<tr>
<td height="30" width="20%"><center>кнопка1</center></td>
```

```
<td width="20%"><center>кнопка2</center></td>
<td width="20%"><center>кнопка3</center></td>
<td width="20%"><center>кнопка4</center></td>
<td width="20%"><center>кнопка5</center></td>
</tr>
<tr>
<td valign="top">содержание:</td>
<td colspan="4"><center>куча текста</center></td>
</tr>
</table>
</center>
</body>
</html>

<html>
<head>
<title>Расстояние между ячейками</title>
</head>
<body>
<center>
<table width="300" height="300" cellspacing="15">
<tr>
<td bgcolor="#c0e4ff" valign="top">1</td>
<td bgcolor="#c5ffa0" valign="top"><center>2</center></td>
<td bgcolor="#c0e4ff" align="right" valign="top">3</td>
</tr>
<tr>
<td bgcolor="#c5ffa0">4</td>
<td bgcolor="#ffa0c5"><center>5</center></td>
<td bgcolor="#c5ffa0" align="right">6</td>
```

```
</tr>
<tr>
<td bgcolor="#c0e4ff" valign="bottom">7</td>
<td bgcolor="#c5ffa0" valign="bottom"><center>8</center></td>
<td bgcolor="#c0e4ff" align="right" valign="bottom">9</td>
</tr>
</table>
</center>
</body>
</html>
```

Ссылки

Существует несколько видов ссылок, а так же "механизмов" перехода по ним. В этой главе постараюсь подробно рассказать о том как прописать ссылки, а так же посвятить в тонкости дела по работе с ними.

Браузеру необходимо знать: точное название документа, путь к документу, и место куда его принести, точнее где его открыть.

На данный момент с помощью блокнота мы создали только один HTML документ у меня он с именем index.html. В этом документе мы попробуем создать ссылку на другой документ, которого у нас еще нет. Так что прежде чем на него ссылаться, его нужно создать, благо Вы это уже умеете.

1. Открываем блокнот.
2. Пишем код на html языке. к примеру страничку с рядом фотографий.
3. Сохраняем его как html страничку в ту же рабочею папку, где уже есть созданный нами первый документ. Давайте, что б не путаться назовем его primer.html, да и первый тоже пожалуй переименуйте в index.html

Теперь знаем, что у Вас два html документа index.html и primer.html и что теперь у Вас есть минимальный набор для дальнейшего обучения.

Текстовые ссылки.

Знакомимся тег <a> (от anchor- якорь), в него можно заключить текст или рисунок, которые станут ссылкой на те или иные документы. Атрибут тега <a> href задаёт имя и путь к документу на который указывает ссылка.

Всё вместе пишется так:

Здесь мои фотки!!

Как Вы наверное поняли primer.html это имя нашего второго html документа, а надпись "Здесь мои фотки!!" это кусочек текста из файла index.html.

По аналогии с рисунками тег <img> путь ссылки к открываемому документу прописывается теми же способами:

Здесь мои фотки!! - Такая запись подразумевает, что в директории, где расположен наш первый html документ есть папка stranica в которой расположен файл primer.html
Здесь мои фотки!! - А это значит, что файл primer.html размещен на уровень выше от документа
Здесь мои фотки!! - документ расположен на сайте www.site.ru.

Ну что давайте попробуем? Ниже приведен пример сразу двух документов в которых прописаны ссылки указывающие друг на друга.

Пример:

Файл index.html:

```
<html>
<head>
<title>Делаем ссылкой кусочек текста</title>
</head>
<body>
<div align=”center”>
<br><br><br><b>Скажи мне, милый ребенок:  в каком ухе у меня жужжит?</b>
```


В правом или левом?

</body>

</html>

Файл primer.html:

<html>

<head>

<title>Перешли по ссылке сюда</title>

</head>

<body>

А вот и не угадал! У меня жужжит в обоих ухах.

</div align=”center”>Ну я так не играю…

</body>

</html>

Из примера видно, что ссылки выделяются цветом, по умолчанию синеньким - ссылка, а красненьким - уже посещенная ссылка, эти цвета можно изменить с помощью уже хорошо известного нам открывающего тега
< body > и его атрибутов.

link - цвет ссылки.

alink - цвет нажатой, активной ссылки.

vlink - цвет посещенной ссылки.

Пишется так:

<body link="#008000" alink="#ff0000" vlink="#ffff00">

Продолжая говорить о цвете текстовой ссылки стоит упомянуть, что при необходимости можно принудительно выделять цветом как всю ссылку, так и отдельные её части (фразы слова буквы) знакомым тегом   и его атрибутом color. Впрочем, это касается не только цвета так же отдельно можно задать размер, стиль и шрифт текста. Но помните, что манипуляции с цветом нужно проводить внутри тега <a>вот здесьа не за бортом, иначе цвет ссылки будет либо по умолчанию, либо так как прописано в теге <body>

Пример:

Файл index.html:

```
<html>
<head>
<title>Радуга</title>
</head>
<body link="#008000" alink="#ff0000" vlink="#ffff00">
<center>
<h3>Посмотрите на фразу которая поможет Вам запомнить места цветов в радуге</h3>
<br>
<a href="primer2.html">
<font size="+1" color=#ff0000>Р</font>
<font size="+2" color=#ff8c40>А</font>
<font size="+3" color=#ffff00>Д</font>
<font size="+3" color=#008000>У</font>
<font size="+2" color=#0000ff>Г</font>
<font size="+1" color=#800080>А</font>
</a>
</center>
</body>
</html>
```

Файл primer.html:

```
<html>
<head>
<title>Радуга</title>
</head>
<body link="#008000" alink="#ff0000" vlink="#ffff00">
<center>
<font size="+3">
<font color=#ff0000>Каждый</font>
<font color=#ff8c40>охотник</font>
<font color=#ffff00>желает</font>
<font color=#008000>знать</font>
<font color=#40caff>где</font>
<font color=#0000ff>сидит</font>
<font color=#800080>фазан</font>
</font>
<br><br><br>
<a href="index.html">вернуться на главную</a>
</center>
</body>
</html>
```

Рисунок ссылка.

Ссылкой может являться не только текст, но и рисунок. Здесь принцип такой же как и в текстовой ссылке, просто вместо текста мы заключаем рисунок который хотим сделать ссылкой.

Вот так:

```
<a href="primer3.html"><img src="knopa.gif"></a>
```

Как при переходе по ссылке открыть документ в новом окне браузера, до этого если Вы обратили внимание он у нас открывался в текущем, что не всегда удобно. Решает эту проблему атрибут target (цель) и его значение _blank.

Пишется так:

открыть в новом окне

Пример:

Файл index.html:

```
<html>
<head>
<title>кнопка</title>
</head>
<body>
<center>
<h1>  Не в коем случае не нажимайте на эту кнопку!!!</h1>
<br>
 <a href=”primer3.html”title=”Не нажимать!!!”target=”_blank”><img src=”graphics/knopa.gif” border=”0” alt=”Не нажимать!!!”></a>
</center>
</body>
</html>
```

Файл primer.html:

```
<html>
<head>
<title>итог…</title>
</head>
<body>
<div align=”center”><font size=”+2”>Ракеты ушли…Америки больше кем</font></div>
</body>
```

</html>

Ссылка на e-mail

Напишите мне письмо - строчка из того примера.

Для того что бы сделать текст или рисунок ссылкой на e-mail - почтовый ящик его нужно заключить в тег <a>, но не простой, а с использованием mailto

Пишем так:

 Напишите мне письмо.

Эта непривычная запись будет говорить что, кликнув по тексту ссылке "Напишите мне письмо" посетитель сайта попадет в свою почтовую программу, которая выдаст ему бланк для отправки письма, где в строчке Кому: уже будет указан нужный нам почтовый ящик mail@yandex.ru

Пример:

```
<html>
<head>
<title>e-mail</title>
</head>
<body>
<center><h12>
<a href=”mailto: mail@yandex.ru ”>Напишите мне письмо.</a>
 </h2></center>
</body>
</html>
```

Списки

В этой главе пойдёт речь о списках, которые могут быть:

- неупорядоченные(маркированные)
- упорядоченные (нумерованные)
- и являться списком определений

Такое вот нестандартное начало главы зато теперь Вам понятно, что есть список в html.

Ну что ж давайте пройдёмся по нашему списку.

Неупорядоченные списки

Неупорядоченный список задаётся тегом <ul>. Любой список состоит из элементов, "пунктов", каждый элемент в свою очередь задаётся тегом после которого собственно и идёт текст нужного нам "пункта". Для тега <li> закрывающий тег необязателен.

Вместе данные теги приобретают следующий вид:

```
<ul>
<li> Пункт первый.
<li> Пункт второй.
<li> Пункт третий.
</ul>
```

Пример неупорядоченного списка:

```
<html>
<head>
<title> неупорядоченный список</title>
</head>
<body>


Купить и доставить домой:
<ul>
<li>2 кг. Клубней картофеля
<li>1 булку ржаного черного хлеба
<li>пачку макарон «спагетти»(длинные не менее 200мм.тонкие)
<li>1 кг. шлифованного риса (сложно объяснить…спросишь…)
<li>1 дес. куриных яиц.
<li>1 пачку 500г. Натрия двууглекислого (Сода)
</ul>
вот
  </body>
```

</html>

Теги <li> и <ul> имеют атрибут type который присуждает элементу списка или же всему списку целиком определённый стиль.

Может иметь одно и трёх значений:

- disk - кружок, диск (по умолчанию)
- circle - полый круг
- square - квадрат

Пример:

<html>

<head>

<title> стили неупорядоченного списка</title>

</head>

<body>

В этом списке каждый элемент имеет свой стиль:

<li type=”disk”> кружок, диск (по умолчанию)

<li type=”circle”> полый круг

<li type=”square”> квадрат

А здесь стиль задан всему списку

</ul type=”circle”>

 Пункт 1

 Пункт 2

 Пункт 3

</body>

</html>

Упорядоченные списки

Упорядоченный или нумерованный список задаётся тегом <ol>, так же как и в неупорядоченном списке элемент списка присуждается тегом<li>.

Построение кода полностью схоже с неупорядоченным списком поэтому сразу пример:

```
<html>
<head>
<title> неупорядоченный список</title>
</head>
<body>
Купить товары в следующем порядке:
<ol>
<li> Сок
  <li> Вода
  <li>Сырок (необязательно)
</ol>
Жду!!!
</body>
 </html>
```

А вот атрибут type в сочетании с упорядоченным списком может иметь следующие значения:

- A - Заглавные буквы
- a - Строчные буквы
- I - Заглавные римские цифры
- i - Строчные римские цифры
- 1 - Арабские цифры (по умолчанию)

Вот пример их применения:

```
<html>
<head>
<title> Стили  неупорядоченного списка</title>
</head>
```

```
<body>
Арабские цифры:
<ol type=”1”>
<li> Во-первых
<li> Во-вторых
  </ol>
Строчные буквы
<ol type=”a”>
<li> Во-первых
<li> Во-вторых
  </ol>
Заглавные буквы
<ol type=”A”>
<li> Во-первых
<li> Во-вторых
<li> В-третьих
 </ol>
Строчные римские цифры
<ol type=”i”>
<li> Во-первых
<li> Во-вторых
<li> В-третьих
<li> В-четвертых
 </ol>
 </body>
</html>
```

В упорядоченном списке есть ещё один атрибут start его числовое значение говорит о том с какого номера следует строить упорядоченный список.

Пример:

```
<html>
<head>
<title> Начало упорядоченного списка</title>
</head>
<body>
<ol type=”1” start=”24”>
<li> Сразу переходим к двадцать четвертому пункту!!
<li> Идем дальше
<li> И дальше
  </ol>
```

Аналогично можно «стартовать» при любом стиле упорядоченного списка

Строчные римские цифры

```
<ol type=”i” start=”8” >
<li>  Сразу переходим к восьмому пункту.
<li> Идем дальше
<li> И дальше
 </ol>
 </body>
 </html>
```

Списки определений

Со списком определений дело обстоит немного иначе нежели чем с уже знакомыми списками. Задаётся данный вид списка тегом <dl>. Пункты списка определений размечаются тегом <dt>, а определения этих пунктов тегом <dd>.

Всё вместе пишется по следующей схеме:

```
<dl>
<dt>
<dd>
</dl>
```

Пример:

```
<html>
<head>
<title>Список определений</title>
</head>
<body>
<dl>
<dt> Слово коса может иметь следующие определения:
<dd> сельскохозяйственный инструмент
<dd> хитрая девичья причёска
<dd> отмель реки
<dt> Слово ключ тоже имеет несколько значений:
<dd> гаечный
<dd> источник, родник
</dl>
</body>
</html>
```

ГЛАВА 2

ОСНОВЫ CSS

CSS (Cascading Style Sheets) - Каскадные таблицы стилей - это свод стилевых описаний, тех или иных HTML тегов (далее элементов HTML), который может быть применён как к отдельному тегу - элементу, так и одновременно ко всем идентичным элементам на всех страницах сайта. CSS по сути своего рода дополнение к HTML, которое значительно расширяет его возможности.

Внедрить в html документ можно тремя способами:

- Написать стилевое описание непосредственно в самом элементе. Такой способ хорош лишь в том случае если таковой элемент один единственный в HTML документе который нуждается в отдельном стилевом описании.
- Написать стилевое описание для всех идентичных элементов HTML документа. Такой способ оправдывает себя, если стиль страницы принципиально отличается от общего дизайна сайта (группы взаимосвязанных страниц).
- Вынести стилевое описание элементов HTML в отдельный файл CSS. Это позволит управлять дизайном всего сайта целиком, каждой страницей сайта в которой указанно обращение к CSS файлу. Этот способ является наиболее эффективным использованием таблицы каскадных стилей.

Атрибут style.

Практически каждый HTML тег имеет атрибут style, который говорит о том, что к этому тегу применяется некое стилевое описание.

Пишется так:

 это параграф с индивидуальным стилем

Всё что будет написано между кавычками атрибута style и будет являться стилевым описанием для данного элемента, в данном случае элемента

Ну например:

 это параграф с индивидуальным стилем

В данном случае мы указали, что этот параграф должен отображаться красным цветом и иметь размер шрифта в 12 пикселей. В последующих главах я подробно расскажу о том что написано в кавычках , сейчас же речь идет о том как применить CSS к какому либо HTML тегу.

По такому же принципу можно указать индивидуальный стиль практически для каждого HTML элемента.

Пример:

</DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd>
<html>
<head>
<title> Атрибут style</title>
</head>
<body style=”background-color: #c5ffa0”>
<h1 style=”color: #0000ff; font-size:18px”> Все о слонах <h1>
<p style=”color: #ff0000; font-size:14px”> На этом сайте вы найдете любую информацию о слонах.
<h2 style=”color: #0000ff; font-size:16px”> Купить слона<h2>
<p style=”color: #ff0000; font-size:14px”> У нас вы можете по выгодным ценам приобрести лучших слонов!!
<h2 style=”color: #0000ff; font-size:16px”> Взять слона на прокат<h2>
<p style=”color: #ff0000; font-size:14px”> Только у нас вы можете взять любых слонов на прокат!!
</body>
</html>

такой способ внедрения CSS хорош лишь в том случае если требуется задать определенный стиль малому числу HTML элементов.

Тег <style>

Для того, что бы описать необходимые элементы одновременно на всей странице в заголовок HTML документа внедряют тег <style> </style>(не путайте с одноименным атрибутом) в котором и происходит описание нужных нам элементов.

Взгляните на пример, ниже к нему будут комментарии.

```
</DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd>
<html>
<head>
<title> Ter style </title>
<style type=”text/css”>
body{background-color: #c5ffa0}
h1 {color: #0000ff; font-size:18px }
h2 {color: #0000ff; font-size:16px }
p   {color: #ff0000; font-size:14px }
</style>
<head>
<body>
<h1> Все о слонах </h1>
<p>На этом сайте вы найдете любую информацию о слонах. </p>
<h2>Купить слона<h2>
<p>У нас вы можете по выгодным ценам приобрести лучших слонов!!
</p>
<h2>Взять слона на прокат<h2>
<p> Только у нас вы можете взять любых слонов на прокат!!</p>
</body>
</html>
```

Как видно из примера мы добились точно такого же результата что и в первом случае только теперь мы не прописываем каждому элементу стиль индивидуально, а вынесли его в "голову" документа тем самым указав что все заголовки <h1>,<h2> - будут синими а параграфы <p> - красными.

Комментарии:

Тег <style> принято внедрять в заголовок HTML документа между тегами <head></head>.

Атрибут тега <style> type - сообщает браузеру, какой синтаксис использовать для правильной интерпретации стилей. Для правильной интерпретации браузерами CSS значение type (MIME тип данных) должно равняться text/css.

Внутри тега <style> </style> идет непосредственное объявление стилей тех или иных HTML элементов согласно следующему синтаксису:

Если в блоке объявления стилей указывается несколько свойств элемента, то они между собой разделяются точкой с запятой.

CSS в отдельном внешнем файле.

Достоинство CSS, возможность выносить все сведения касающиеся дизайна сайта в отдельный внешний файл.

Итак, открываем блокнот (или другой редактор) и пишем в нем следующий текст:

```
body {background-color: #c5ffa0}
a {color: #000060; font-weight: bold;}
```

```
a:hover {color: #ff0000; font-weight: bold; text-decoration;none }
h1   {color: #0000ff; font-size:18px }
h2  {color: #ff00ff; font-size:16px }
p {color: #600000; font-size:14px }
```

Далее сохраняем этот небольшой файлик с расширением *.css (обычно файл со стилями называют style.css).

Файл со стилевым описанием создан! Теперь осталось заставить нужные страницы нашего сайта черпать информацию с этого файла.

Делается это с помощью тега <link> (связь). Тег <link> многоцелевой и служит для "связывания" HTML документа с дополнительными внешними файлами, обеспечивающими его должную работу. Тег <link> является своего рода ссылкой, только предназначенной не для пользователей, а для программ обозревателей (браузеров). Так как <link> несёт в себе исключительно служебную информацию он располагается в заголовке HTML документа между тегами <head></head> и не выводится браузерами на экран.

Тег <link> имеет атрибуты:

href - Путь к файлу.

rel - Определяет отношения между текущим документом и файлом, на который делается ссылка.

- shortcut icon - Определяет, что подключаемый файл является иконкой.
- stylesheet - Определяет, что подключаемый файл содержит таблицу стилей.
- application/rss+xml - Файл в формате XML для описания ленты новостей.

type - MIME тип данных подключаемого файла.

Так как мы подключаем в качестве внешнего файла каскадную таблицу стилей, то наша служебная ссылка приобретает следующий вид:

```
<link rel="stylesheet" href="mystyle.css" type="text/css">
```

Атрибуту rel присваиваем значение stylesheet так как подключаем в качестве внешнего файла каскадную таблицу стилей, указываем путь к файлу css (в этом примере файл называется mystyle.css и лежит рядом с документом HTML в котором прописывается данная ссылка) так же указываем, что данный файл текстовый и содержит в себе стилевое описаниеtype="text/css"

Теперь вставляем эту строчку в заголовки страниц нашего сайта Пример:

Файл mystyle.css

```
body {background-color: #c5ffa0}
a {color: #000060; font-weight: bold;}
a:hover {color: #ff0000; font-weight: bold; text-decoration;none}
h1   {color: #0000ff; font-size:18px }
h2  {color: #ff00ff; font-size:16px }
p {color: #600000; font-size:14px }
```

Файл index.html

```
</DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd>
<html>
<head>
<title> Каскадная таблица стилей </title>
<link rel=”stylesheet” href=”mystyle.css” type=”text/css”>
<head>
<body>
<h2>Меню:</h1>
<a href=”index.html”>Все о слонах. </a>
<a href=”elephant.html”>Купить слона</a>
<a href=” elephant1.html”> Взять слона на прокат</a>
<hr>
<h1>Все о слонах</h1>
<p> На этом сайте вы найдете любую информацию о слонах. </p>
```

```
  </body>
</html>
```

Файл elephant.html

```
</DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd>
<html>
<head>
<title> Каскадная таблица стилей </title>
<link rel=”stylesheet” href=”mystyle.css” type=”text/css”>
<head>
<body>
<h2>Меню:</h1>
<a href=”index.html”>Все о слонах. </a>
<a href=”elephant.html”>Купить слона</a>
<a href=” elephant1.html”> Взять слона на прокат</a>
<hr>
<h1>Все о слонах</h1>
<p> На этом сайте вы найдете любую информацию о слонах. </p>
  </body>
</html>
```

Файл elephant.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>каскадная таблица стилей</title>
<link rel="stylesheet" href="mystyle.css" type="text/css">
```

```
</head>
<body>
<h2>Меню:</h2>
<a href="index.html">Всё о слонах.</a>
<a href="elephant.html">Купить слона.</a>
<a href="elephant1.html">Взять слона на прокат.</a>
<hr>
<h1>Купить слона</h1>
<p>У нас Вы можете по выгодным ценам приобрести лучших
слонов!!</p>
</body>
</html>
```

Файл elephant1.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>каскадная таблица стилей</title>
<link rel="stylesheet" href="mystyle.css" type="text/css">
</head>
<body>
<h2>Меню:</h2>
<a href="index.html">Всё о слонах.</a>
<a href="elephant.html">Купить слона.</a>
<a href="elephant1.html">Взять слона на прокат.</a>
<hr>
<h1>Взять слона на прокат</h1>
<p>Только у нас Вы можете взять любых слонов на прокат!!</p>
```

```
</body>
</html>
```

Стиль шрифта

Свойство font-style, в зависимости от выбранного значения, определяет стиль шрифта.

Шрифт может иметь следующие стили:

- normal - обычный (по умолчанию)
- italic - курсив
- oblique – наклонный

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Стиль шрифта</title>
</head>
<body>
<p style="font-style: italic">это курсив</p>
<p style="font-style: oblique">а это наклонный текст</p>
<p style="font-style: normal">И чем спрашивается, они отличаются?</p>
</body>
</html>
```

Чем отличается курсив от наклонного текста? Курсив - это своего рода шрифт взятый из библиотеки шрифтов, а наклонный текст - это результат работы алгоритма, где каждый символ слегка наклоняется в правую сторону.

Начертание шрифта

Весьма интересное свойство шрифта font-variant позволяет делать строчные буквы заглавными и уменьшенными.

Значения:

- normal - нормальный (по умолчанию)

- small-caps - все буквы заглавные и уменьшенные

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Все буквы заглавные</title>
</head>
<body>
<p style="font-variant: small-caps">Купи слона! </p>
</body>
</html>
```

Размер шрифта

Свойство CSS font-size - определяет размер шрифта.

Размер шрифта может быть задан в процентах или пикселях и любых других допустимых единицах измерения CSS, а так же абсолютным или относительным значением.

значения абсолютного размера шрифта:

- xx-small - очень очень маленький
- x-small - очень маленький
- small - маленький
- medium - средний
- large - большой
- x-large - очень большой
- xx-large - очень очень большой

значения относительного размера шрифта:

- larger - больше чем размер шрифта родительского элемента
- smaller - меньше чем размер шрифта родительского элемента
- Пример:

```
</DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN”
http://www.w3.org/TR/html4/loose.dtd>
<html>
<head>
<title> Размер шрифта</title>
</head>
<body>
 <div style=”font-size: 18px; background-color: #ecfef2; border: 5px double #245404”>
<p> Размер шрифта родительского элемента (блока DIV) равен 18 пикселям </p>
<p style=”font-size: larger” > Этот шрифт будет крупнее относительно элемента родителя  </p>
<p style=”font-size: smaller”> А этот шрифт будет мельче относительно элемента родителя</p>
</div>
<p style=”font-size: 14 px”> В блоке ниже размер шрифта элемента родителя огромен(60 пунктов), однако дочерние параграфы расположенные в нём имеют собственное абсолютное значение шрифта и к размеру шрифта элемента родителя никак не привязаны. </p>
<div style=”font-size: 60 pt; background-color: #ecfef2; border: 5px double #245404”>
<p style=”font-size: xx-small”> xx-small – очень очень маленький </p>
<p style=”font-size: x-small”> x-small – очень маленький </p>
<p style=”font-size: small”> small –маленький </p>
<p style=”font-size: medium”> medium – средний </p>
<p style=”font-size: large”> large –большой </p>
<p style=”font-size: x-large”>x-large – очень большой </p>
<p style=”font-size: xx-large”> xx-large – очень очень большой </p>
</div>
```

</body>
</html>

Жирность шрифта

Свойство font-weight - определяет жирность шрифта. Насыщенность шрифта может быть задана относительно шрифта элемента родителя с помощью следующих значений:

- normal - обычный шрифт
- bold - полужирный шрифт
- bolder - жирный шрифт
- lighter - тонкий шрифт

А также выражается в условном числовом значении от 100 до 900 с шагом 100 (100, 200, 300.. 900) где значение 100 тонкий шрифт, а 900 - сверх жирный.

Цвет и фон.

В этой главе мы поговорим о том, как с помощью CSS присвоить цвет элементу и его фону, а так же о том, как использовать рисунок в качестве фона элемента и управлять его положением.

Перед тем как перейти непосредственно к обучению, проведу краткий экскурс на тему: "Цвета в Интернете"

Цвет в CSS может быть задан тремя методами:

- Именным значением, например: red - красный.
- Значением цвета RGB, например: RGB(255,0,0) - опять таки красный.
- Шестнадцатеричным значением цвета RGB, например: #ff0000 - всё тот же красный.

С именным значением цвета всё понятно black - черный, green - зелёный, olive - оливковый и т.д. (полную палитру базовых красок, т.е. цветов для которых зарезервированы именные значения, смотрите здесь.)

Однако по понятным причинам не для всех оттенков цветов зарезервировано индивидуальное имя. Когда возникает необходимость в использовании какого либо "нестандартного" цвета необходимо определить его

значение RGB, (Red, Green , Blue) сочетание красного, зеленого и синего цвета в числовом выражении. Каждый оттенок из основных цветов в системе RGB может выражаться в числе от 0 до 255.

Например, черный цвет будет иметь значение 0,0,0 то есть отсутствие всякого цвета. белый - значение255,255,255 теоретически если смешать основные цвета должен получится белый, а вот например классический синий цвет имеет значение 0,0,255 то есть на "мольберте" присутствует только синий. На рисунке наглядно показано, что происходит с красками если их смешать, так смешивая оттенки основных цветов можно добиться любого цвета из видимого спектра.

Однако в большинстве случаев "веб краски" имеют шестнадцатеричное выражение десятичного значения RGB.

В шестнадцатеричном исчислении цифры от 10 до 15 заменены латинскими буквами и числовой ряд приобретает следующий вид:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Числа больше 15ти в шестнадцатеричной системе образуются путём объединения двух и более чисел в одно. Так например, числу 255 в десятичной системе соответствует число FF в шестнадцатеричной системе.

Значит, для того чтобы выразить нужный оттенок в шестнадцатеричном виде, нам понадобиться три пары чисел, где первая пара - значение красного цвета, вторая пара значение зелёного и третья пара синего цвета. Так, например, тот же классический синий в шестнадцатеричном выражении будет выглядеть так: #0000FF. Знак решётки перед числом ставится для указания того что данное число является шестнадцатеричным, например в числе #808000 нет латинских букв однако со знаком решётки понятно что оно шестнадцатеричное и выражает собой оливковый цвет.

И еще. выражая цвет в шестнадцатеричном виде можно обойтись тремя числами, которые затем будут дублироваться, например запись #DAFбудет сокращённой формой #DDAAFF.

Цвет элемента.

Для того, что бы перекрасить текст, какого либо, элемента в нужный нам цвет необходимо воспользоваться свойством color и присвоив ему нужное значение - собственно цвет.

Как уже сказано выше цвет в CSS может быть задан следующими методами:

- #ff0000 - шестнадцатеричное значение цвета RGB.
- red - именное значение цвета.
- RGB(255,0,0) - значение цвета RGB.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title> Цвет элемента  </title>
 </head>
<body>
<div style="color: red">Блок 1 </div>
<div style="color: #ff0000">Блок 2 </div>
<div style="color: RGB(255,0,0) ">Блок 3 </div>
</body>
</html>
```

Цвет фона элемента.

А вот свойство background-color - определяет цвет фона элемента.

Цвет фона может иметь следующие значения:

- #ff0000 - шестнадцатеричное значение цвета RGB.
- red - именное значение цвета.
- RGB(255,0,0) - значение цвета RGB.
- transparent - прозрачный фон. (по умолчанию)

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Цвет фона элемента</title>
</head>
<body style="background-color: #fffacd">
<div style="background-color: white">Белый блок</div>
<div style="background-color: #0000ff">Синий блок</div>
<div style="background-color: RGB(255,0,0)">Красный блок</div>
<div style="background-color: transparent">Прозрачный блок</div>
</body>
</html>
```

Фоновое изображение.

Для любого элемента можно присвоить фоновое изображение с помощью CSS свойства: background-image.

Возможные значения background-image:

- url - путь к файлу с изображением.
- none - изображение отсутствует. (по умолчанию)

Для того чтобы сделать некую картинку фоном для элемента необходимо указать к ней путь согласно следующего синтаксиса url(путь к файлу/имя файла). Путь к файлу указывается в том случае, если рисунок находится в другой папке.

В примере ниже в качестве основного фона (элемент body) используется одно графическое изображение, а для блока div другое, возможность

использования различных фоновых изображений для разных элементов страницы позволяет решать практически любые дизайнерские задумки.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Фоновое изображение</title>
<style type="text/css">
body{
background-image: url(fon.jpg);
}
div{
background-image: url(fon1.gif);
border: 5px double #245404;
height: 250px;
}
p{
text-align: center;
color: #008040;
font: bold 24px Arial;
}
</style>
</head>
<body>
<p>Страница с фоновым изображением</p>
<div><p>Блок с фоновым изображением</p></div>
</body>
</html>
```

Границы элемента.

В этой главе мы поговорим о том, как сделать с помощью CSS рамку - бордюр, вокруг того или иного элемента. В HTML эта задача лежала на плечах атрибута border, однако его можно было применить далеко не к каждому тегу (элементу) да и не всегда он мог решить ту или иную дизайнерскую задумку.

В CSS эту задачу берёт на себя одноимённое базовое свойство border и значительно расширяет круг возможностей при работе со стилем границы любого(!) элемента выводимого на экран.

Стиль границы.

Если в HTML бордюр мог быть только в виде сплошной линии вокруг элемента, то в CSS это уже достаточно широкий набор возможных стилей рамок.

Свойство border-style может присваивать элементу один из ниже перечисленных стилей границы.

- none - граница отсутствует (по умолчанию).
- dotted - граница из ряда точек.
- dashed - пунктирная граница.
- solid - сплошная граница
- double - двойная граница
- groove - граница "бороздка"
- ridge - граница "гребень"
- inset - вдавленная граница
- outset - выдавленная граница

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Стиль границы</title>
<style type="text/css">
```

```
p {
background-color: #f5f5f5;
text-align: center;
}
</style>
</head>
<body>
<p style="border-style: none;">граница отсутствует</p>
<p style="border-style: dotted;">граница из ряда точек</p>
<p style="border-style: dashed;">пунктирная граница</p>
<p style="border-style: solid;">сплошная граница</p>
<p style="border-style: double;">двойная граница</p>
<p style="border-style: groove;">граница "бороздка"</p>
<p style="border-style: ridge;">граница "гребень"</p>
<p style="border-style: inset;">вдавленная граница</p>
<p style="border-style: outset;">выдавленная граница</p>
</body>
</html>
```

Стиль бордюра может быть задан как для всех сторон элемента одновременно, так и для каждой его стороны отдельно в зависимости от того, сколько значений присвоено свойству border- style. Таковых значений может быть от одного до четырёх по числу сторон элемента.

В каждом из четырёх случаев действуют свои "правила" по присуждению стиля рамки той или иной стороне элемента, которые приведены в таблице ниже:

Число значений	Результат
1	Пример: div {border-style: solid;} • Первое значение - Устанавливает единый стиль

	бордюра для всех сторон элемента.
2	Пример: div {border-style: solid double;} • Первое значение - Устанавливает стиль верхней и нижней границы элемента. • Второе значение - Устанавливает стиль левой и правой границы элемента.
3	Пример: div {border-style: solid double dashed;} • Первое значение - Устанавливает стиль верхней границы элемента. • Второе значение - Устанавливает стиль левой и правой границы элемента. • Третье значение - Устанавливает стиль нижней границы элемента.
4	Пример: div {border-style: solid double dashed ridge;} • Первое значение - Устанавливает стиль верхней границы элемента. • Второе значение - Устанавливает стиль правой границы элемента. • Третье значение - Устанавливает стиль нижней границы элемента. • Четвёртое значение - Устанавливает стиль левой границы элемента.

Толщина границы.

Свойство border-width - устанавливает ширину границы элемента.

Ширина бордюра может быть заданна с помощью следующих аргументов:

- thin - тонкая граница
- medium - средняя толщина границы

- thick - толстая граница

А также в пикселях или любых других единицах измерения принятых в CSS.

По аналогии со стилем, толщина бордюра тоже может иметь от одного до четырёх значений и в каждом случае устанавливает ёе для тех или иных сторон бордюра как показано в таблице ниже.

Число значений	Результат
1	Пример: div {border-style: solid; border-width: 1px;} • Первое значение - Устанавливает единую толщину бордюра со всех сторон.
2	Пример: div {border-style: solid; border-width: 1px 4px;} • Первое значение - Устанавливает толщину верхней и нижней границы элемента. • Второе значение - Устанавливает толщину левой и правой границы элемента.
3	Пример: div {border-style: solid; border-width: 1px 4px 7px;} • Первое значение - Устанавливает толщину верхней границы элемента. • Второе значение - Устанавливает толщину левой и правой границы элемента. • Третье значение - Устанавливает толщину нижней границы элемента.
4	Пример: div {border-style: solid; border-width: 1px 4px 7px 5px;} • Первое значение - Устанавливает толщину верхней

	границы элемента. • Второе значение - Устанавливает толщину правой границы элемента. • Третье значение - Устанавливает толщину нижней границы элемента. • Четвёртое значение - Устанавливает толщину левой границы элемента.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Толщина границы</title>
</head>
<body>
<div style="border-style: solid; border-width: 3px 1px 10px 4px">
Толщина границ данного блока:
<ul>
<li>Сверху 3 пикселя
<li>Справа 1 пиксель
<li>Снизу 10 пикселей
<li>Слева 4 пикселя
</ul>
</div>
<br><br>
<div style="border-style: double; border-width: thick">В этом блоке границы со всех сторон одинаково толстые</div>
</body>
</html>
```

Цвет рамки или её сторон по отдельности определяется свойством border-color.

Цвет бордюра может иметь следующие значения:

- #ff0000 - шестнадцатеричное значение цвета RGB.
- red - именное значение цвета.
- RGB(255,0,0) - значение цвета RGB.
- transparent - прозрачная граница.

Ну и так же как и в случаях с толщиной и стилем, цвет бордюра тоже может иметь от одного до четырёх цветовых значений при каждом "раскладе" окрашивая нужные стороны бордюра как показано в таблице ниже.

Число значений	Результат
1	Пример: div {border-style: solid; border-width: 3px; border-color: #008000;} • Первое значение - Устанавливает единый цвет бордюра со всех сторон.
2	Пример: div {border-style: solid; border-width: 3px; border-color: #008000 #0000ff;} • Первое значение - Устанавливает цвет верхней и нижней границы элемента. • Второе значение - Устанавливает цвет левой и правой границы элемента.
3	Пример: div {border-style: solid; border-width: 3px; border-color: #008000 #0000ff #ff0000;} • Первое значение - Устанавливает цвет верхней границы элемента. • Второе значение - Устанавливает цвет левой и правой границы элемента.

	• Третье значение - Устанавливает цвет нижней границы элемента.
4	Пример: div {border-style: solid; border-width: 3px; border-color: #008000 #0000ff #ff0000 #ffff00;} • Первое значение - Устанавливает цвет верхней границы элемента. • Второе значение - Устанавливает цвет правой границы элемента. • Третье значение - Устанавливает цвет нижней границы элемента. • Четвёртое значение - Устанавливает цвет левой границы элемента.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Цвет границы</title>
</head>
<body>
<div style="border-style: solid; border-width: 10px; border-color: #ff0000 #ffff00 #00ff00 #0000ff;">
<p style="border-style: double; border-width: 5px; border-color: #008000;">Зелёный</p>
<p style="border-style: double; border-width: 5px; border-color: red;">Красный</p>
<p style="border-style: double; border-width: 5px; border-color: rgb(0,0,255);">Синий</p>
</div>
```

```
</body>
</html>
```

Границы справа слева сверху и снизу отдельно.

Для того, что бы определить стиль, цвет и ширину бордюра для одной из сторон элемента, пользуйтесь свойствами border-bottom, border-left, border-right, border-top и их дочерними "коллегами по цеху" список которых приведён ниже:

- border-bottom-color - Устанавливает цвет нижней границы элемента.
- border-bottom - Определяет стиль, цвет и ширину нижней границы элемента
- border-bottom-style - Определяет стиль нижней границы элемента.
- border-bottom-width - Определяет ширину нижней границы элемента.

border-left - Определяет стиль, цвет и ширину левой границы элемента.

- border-left-color - Устанавливает цвет левой границы элемента.
- border-left-style - Определяет стиль левой границы элемента.
- border-left-width - Определяет ширину левой границы элемента.
- border-right - Определяет стиль, цвет и ширину правой границы элемента.
- border-right-color - Устанавливает цвет правой границы элемента.
- border-right-style - Определяет стиль правой границы элемента.
- border-right-width - Определяет ширину правой границы элемента.
- border-top-color - Устанавливает цвет верхней границы элемента.
- border-top-style - Определяет стиль верхней границы элемента.
- border-top-width - Определяет ширину верхней границы элемента.

Не буду описывать каждый из них, думаю и так понятно, что дело обстоит, так же как и с их сородичами, свойствами border-style, border-width и border-color, кроме того факта, что в данном случае свойства указываются для одной из сторон границы элемента.

Приведу пример:

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

<html>

<head>

<title>Граница слева</title>

<style type="text/css">

div{

border-left: 10px solid #008000;

background: #c6f2de;

}

</style>

</head>

<body>

В этом примере у контейнера div мы обозначили только его левую границу с помощью свойства border-left написав:

div{

border-left: 10px solid #008000;

background: #c6f2de;
 }

Такого же результата можно было бы добиться прописывая свойства стиля, толщины и цвета для левой границы элемента отдельно.

Это выглядело бы вот так:

div{

border-left-color: #008000;

border-left-style: solid;

border-left-width: 10px;

background: #c6f2de;

}

Кстати Вам этот блок ничего не напоминает? :)

```
</div>
</body>
</html>
```

Border

Свойство border - базовый атрибут одновременно определяет стиль, цвет и толщину границы элемента.

Так как атрибут border является базовым, значения родственных свойств указываются в любом порядке через пробел.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>border</title>
<style type="text/css">
div{
border: RGB(25, 125, 25) 6px ridge;
}
</style>
</head>
<body>
<div>
<h3>А знаете ли Вы что:</h3>
<p>Слон - символ положительного характера - используется в Азии как царское верховное животное и высоко ценится за ум и хитрость.</p>
.. .. ..
</div>
</body>
</html>
```

Однако если Вы хотите присвоить разные свойства различным сторонам границы элемента или только одной из них, пользуйтесь свойствами border-bottom, border-left, border-right, border-top.

Границы таблицы.

Свойство CSS border-collapse определяет стиль отображения границ таблицы.

По умолчанию каждая ячейка таблицы имеет собственную рамку (ну если конечно использован атрибут HTML border или одноимённое свойство CSS), так вот в местах соприкосновения ячеек образуется двойная линия, border-collapse заставляет браузер анализировать таковые места и поступать с ними согласно присвоенному значению данному свойству.

Внешний вид границ таблицы может принимать следующий вид:

- separate - ячейки таблицы отделены друг от друга (по умолчанию).
- collapse - ячейки таблицы не имеют промежутков между собой.
- inherit - свойства наследуются у родителя элемента. (работает далеко не во всех браузерах.)

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Стиль таблицы</title>
</head>
<body>
<table cellpadding="5" border="5">
<caption>Таблица с бордюром по умолчанию</caption>
<tr>
<td>ячейка</td><td>ячейка</td><td>ячейка</td><td>ячейка</td>
</tr>
```

```
<tr>
<td>ячейка</td><td>ячейка</td><td>ячейка</td><td>ячейка</td>
</tr>
</table>
<hr>
<table cellpadding="5" border="5" style="border-collapse: collapse">
<caption>А эта таблица использует свойство CSS border-collapse с значением collapse</caption>
<tr>
<td>ячейка</td><td>ячейка</td><td>ячейка</td><td>ячейка</td>
</tr>
<tr>
<td>ячейка</td><td>ячейка</td><td>ячейка</td><td>ячейка</td>
</tr>
</table>
</body>
</html>
```

Классы и идентификаторы.

Классы CSS.

Начнём с классов.

Как присвоить элементу или группе идентичных элементов индивидуальный стиль, отличный от основного, уже указанного в стилевом описании документа? Не знаю задавались Вы этим вопросом или нет, но рано или поздно на него необходимо найти ответ.

Итак. предположим в файле CSS к элементу <p> у нас применён следующий стиль:

```
p {color: #0000ff; font-size:14px}
```

И все вроде бы хорошо. все параграфы синенькие и размер у них 14px, но нам надо сделать так чтобы некоторые из этих параграфов были розовые!

На помощь приходят классы.

Для того чтобы выделить некоторые из параграфов розовым цветом, необходимо присвоить элементу определённое имя и вывести его тем самым в класс, в некую нестандартную, для страницы или сайта в целом, категорию.

Ну что давайте попробуем? Делается это так:

p.rose {color: #ff00ff; font: italic 16px Arial}

Поясню p - это элемент HTML (селектор) в данном случае наш параграф, .rose - это индивидуальное имя класса которое мы сами выдумали, оно может быть любым необязательно rose-розовый, точка между селектором и именем класса есть дань уважения к синтаксису принятому в CSS - теперь браузер поймет что данный элемент p выведен в класс rose.

Ну что ж имя мы присвоили теперь нам необходимо в документе HTML указать теги (в нашем случае теги) которым необходим индивидуальный стиль. Делается это с помощью атрибута class.

Вот так:

Этот параграф использует имя класса rose и тем самым выделяется из основной массы

Пример:

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

<html>

<head>

<title>Внедрение класса</title>

<style type="text/css">

body {background-color: #c5ffa0}

p {color: #0000ff; font-size:14px}

p.rose {color: #ff00ff; font: italic 16px Arial}

```
</style>
</head>
<body>
<p>На этом сайте Вы найдёте любую информацию о слонах.</p>
<p>У нас Вы можете по выгодным ценам приобрести лучших слонов!!</p>
<p>Только у нас Вы можете взять любых слонов на прокат!!</p>
<p class="rose">Специальное предложение для девушек! Розовые слоны!! только у нас!!!</p>
</body>
</html>
```

В данном примере класс "rose" может быть присвоен только параграфу - элементу p. Для того чтобы данное стилевое описание могло распространятся на все элементы, в файле CSS (или между тегами <style></style> в заголовке документа) элемент явно не указывается и синтаксис приобретает следующий вид:

```
.rose {color: #ff00ff}
```

Теперь указав в любом элементе class="rose" он примет стиль данного класса.

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Классы</title>
<style type="text/css">
body {background-color: #c5ffa0}
h1 {color: #0000ff; font-size:22px}
p {color: #008000; font: italic 16px Arial}
```

```
span {color: #0080ff; font-size:12px}
.rose {color: #ff00ff}
</style>
</head>
<body>
<h1>Это заголовок с основным стилем CSS</h1>
<h1 class="rose">А этот заголовок использует class="rose"</h1>
<hr>
<a href="#" title="Обыкновенная ссылка">Это ссылка по умолчанию</a><br>
<a href="#" title="Розовая ссылка" class="rose">А эта ссылка использует class="rose"</a><br>
<hr>
<span>Этот строковый блок использует основной стиль</span><br>
<span class="rose">А этот класс rose</span>
<hr>
<p>Параграф без указания класса</p>
<p class="rose">Параграф с указанием класса</p>
</body>
</html>
```

Обратите внимание на тот факт, что недостающие описания стиля выведенного в отдельный класс, элементы черпают из основного стилевого описания или же берут из свойств элемента "по умолчанию".

Например, заголовок <h1 class="rose"> был синим, а стал розовым, но при этом сохранил свой размер 22 пикселя, так как в нашем классе rose никаких разговоров о размере шрифта не шло. мы лишь "договорились" с браузером, что элементы из группы rose будут розовыми.

Идентификаторы

Идентификаторы они же id селекторы, весьма схожи с классами, с тем лишь отличием, что идентификатор может иметь одно единственное уникальное имя во всем документе. Идентификаторы, как правило, применяются в том случае, если возникает необходимость управлять стилем элемента динамически с помощью скрипта, обращаясь к его индивидуальному имени.

В файле CSS имя указывается со знаком решётки в его начале.

Вот так, например:

```
#block {color: #ff00ff; font: italic 16px Arial}
```

А к нужному элементу добавляется атрибут id="block" например

```
<p id="block">Параграф с идентификатором</p>
```

Вот пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Идентификатор</title>
<style type="text/css">
body {background-color: #c5ffa0}
p {color: #0000ff; font-size:14px}
#rose {color: #ff00ff; font: italic 16px Arial}
</style>
</head>
<body>
<p>Это обыкновенный параграф</p>
<p id="rose">А этот параграф уникальный</p>
</body>
</html>
```

Ну так в чем же отличие между классом и идентификатором?

Пример:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Идентификаторы и скрипты</title>
<script>
function show_hide(id){
var item = document.getElementById(id);
if (item.style.display == 'none') {item.style.display = 'block';}
else item.style.display = 'none';
}
</script>
</head>
<body>
<div id="block" style="display:none">
<h2 style="color: #ff00ff">А вот и я!!</h2>
<img src="rosemammoth.gif">
</div>
<a href="javascript:show_hide('block')" title="Развернуть/Свернуть"
style="color: #ff00ff">Нажми на меня!!</a>
<hr>
<div id="block1" style="display:none">
<h2 style="color: #0000ff">А здесь я!!</h2>
<img src="mammoth.jpg">
</div>
<a href="javascript:show_hide('block1')" title="Развернуть/Свернуть"
style="color: #0000ff">И на меня нажми!!</a>
</body>
```

</html>

Курсивом, в данном примере, выделен скрипт, который может динамически обрабатывать блоки с уникальными именами "block" и"block1" (скрывать и показывать его по нажатии на ссылку), и хотя пока Вам, думаю, мало, что понятно из выше написанного, но цель данного примера показать, как скрипт может обращаться к блоку через атрибут id. А вот с помощью классов мы бы не достигли такого результата.

Заключение

Как выложить сайт в Интернет? Для того что бы Ваш сайт стал доступен много миллионной публике сети Интернет Вам необходимо сделать две вещи:

- Присвоить ему индивидуальное имя - домен.
- Разместить сайт на каком либо сервере - (воспользоваться услугой хостинга)

Прежде чем вдаваться в подробности касающейся публикации сайтов давайте осмыслим следующие понятия:

Web - сервер: По сути web - сервер это обычный компьютер подключенный к Интернету 24 четыре часа в сутки на котором установлено специальное программное обеспечение. Впрочем отличия от обыкновенного домашнего компьютера у него есть и этих отличий довольно много. Ну во-первых нормальный сервер должен бесперебойно без перезагрузок и выключений работать в течении многих дней а это значит он должен иметь бесперебойное и альтернативное питание, надёжное "железо" и его качественное обслуживание. Во - вторых он должен обеспечивать хранение огромных объёмов информации, обрабатывать тысячи запросов пользователей, что в свою очередь ставит высокие требования к производительности данного компьютера. В - третьих он должен быть надежно защищён от нападок вирусов, хакерских атак, да от чего угодно будь то даже нашествие грызунов. (ничего смешного. мыши порой работают почище всяких там хакеров) следовательно этот компьютер должен иметь систему резервного копирования.

Все выше перечисленные требования к web - серверу подводят нас тому, что нам придётся воспользоваться услугой хостинга.

Хостинг: Платная или бесплатная услуга размещения сайта (информации) на каком либо профессиональном web - сервере. Ниже поговорим более подробно о том как воспользоваться услугой хостинга.

Web - узел: Ваше место (папка) на сервере к содержанию которой Вы будете иметь доступ. Как правило администраторы сервера не предоставляют информации о том где именно она расположена в общем дереве каталогов

сервера. Впрочем эта информация нам и не пригодится. Нам нужна только наша папка в которую мы и разместим сайт в том виде как и в рабочей папке на своём компьютере, построив в ней своё собственное дерево каталогов.

Доменное имя (домен): Индивидуальное неповторимое имя Вашего сайта, например www.site.ru Для того что бы получить собственный домен необходимо его зарегистрировать у лицензированного регистратора доменных имен. Опять таки платно. или бесплатно, но это будет домен третьего уровня. (ниже расскажу что такое уровни домена)

Учебно-методическое пособие

Еникеев Арслан Ильясович

Степанова Элина Радиковна

СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ. ОСНОВЫ WEB-ПРОГРАММИРОВАНИЯ