Содержание

Введение

В 80-е годы прошлого века термин «гипертекст» в филологическом смысле употреблял известный структуралист-нарратолог Жерар Женетт в своей книге «Палимпсесты»[1] , рассуждая о различных видах интертекстуальных связей. Жерар Женнет разработал практическую модель исследования межтекстовых отношений, уделяя основное внимание не частным текстуальным связям, а произведению как целостной структуре. Он предложил собственную классификацию разных типов взаимодействия текстов. Однако термин «гипертекст» в рамках концепции Женетта имел более узкое значение, чем в компьютерных разработках, где он становится ключевым словом.

Существует множество определений гипертекста. Самое простое из них можно найти почти в любом руководстве по веб-дизайну: Гипертекст — это текст, связанный ссылками с другими текстами.

Тед Нельсон определял гипертекст следующим образом: «Под гипертекстом я понимаю не последовательное сочинение (non-sequential writing) а текст, который разветвляется и позволяет читателю выбирать <…>. Проще говоря, это ряд кусков текста (a series of text chunks), соединенных линками, предлагающими читателю различные пути« .[2]

Словарь культуры XX века В.Руднева дает следующее определение гипертекста: Гипертекст — текст, устроенный таким образом, что он превращается в систему, иерархию текстов, одновременно составляя единство и множество текстов. [3]

Для полноты картины приведем и другие достаточно типичные дефиниции гипертекста собранные из разных источников.

1. Гипертекст — это соединение смысловой структуры, структуры внутренних связей некоего содержания, и технической среды, технических средств, дающих возможность человеку осваивать структуру смысловых связей, осуществлять переходы между взаимосвязанными элементами [4].

2. Механизм, заключающийся в возможности связать отрывки текста, переходить от одного к другому, называется гипертекстом или нелинейным текстом [5].

3. Надтекст, некая единица информации, частями которой являются тексты и/или текст, части которого имеют «сверхсвязи», то есть соединены друг с другом не линейным отношением в одномерном пространстве (отношением следования как в обычном тексте естественного языка), а множеством различных отношений, представляемых в многомерном пространстве. В гипертексте отсутствуют заранее заданные ограничения на характер связей (сеть) [6].

4. Гипертекст можно определить как нелинейную документацию, которая ветвится и взаимосвязывается, позволяя читателю исследовать содержащуюся в ней информацию в последовательности, которую он сам выбирает. Гипертекст позволяет связывать текст, аудио, фотографии, чертежи, карты, движущиеся картинки и другие формы информации в осмысленное целое, доступ к которому может осуществляться при помощи системы индексации, ориентированной на конкретные идеи, а не на конкретные слова в тексте [7].

5. Гипертекст — это текст, организованный таким образом, что из некоторых его элементов возможен алеаторический (по выбору читателя) переход сразу на несколько других элементов либо возможны ссылки на другие элементы этого же текста или других текстов[8].

6. Гипертекст — это представление информации как связанной (linked) сети гнезд (nodes), в которых читатели свободны прокладывать путь (navigate) нелинейным образом. Он допускает возможность множественности авторов, размывание функций автора и читателя, расширение работы с нечеткими границами и множественность путей чтения» [9].

Почти о том же писал и Р.Барт:

«…текст пронизан сетью бесчисленных, переплетающихся между собой внутренних ходов, не имеющих друг над другом власти; он являет собой галактику означающих, а не структуру означаемых; у него нет начала, он обратим; в него можно вступить через множество входов, ни один из которых нельзя признать главным; вереница мобилизуемых им кодов теряется где-то в бесконечной дали, они «неразрешимы» (их смысл не подчинен принципу разрешимости, так что любое решение будет случайным, как при броске игральных костей); этим сугубо множественным текстом способны завладеть различные смысловые системы, однако их круг не замкнут, ибо мера таких систем — бесконечность самого языка« [10].

Глава 1. Об истории языков гипертекстовой разметки

Изначально слово разметка, как правило, использовалось для описаний аннотаций или других обозначений внутри текста, которые предназначались для указаний составителю документа или, как его иногда называют, "верстальщику" того, как именно конкретное место должно быть напечатано. Подобные способы могут включать в себя подчеркивание волнистой чертой, обозначающее курсив, какие-либо специальные значки для пропуска отдельных фраз или их печати конкретным шрифтом, и так далее. Когда с течением времени форматирование и печать стали автоматизированными, этот термин уже охватывал все виды специальных кодов разметки, которые вставлялись в электронные текстовые документы для управления форматированием, печатью либо другой обработкой.

Под языком разметки, таким образом, понимают набор соглашений о принципах форматирования, которые применяются для кодирования текстовых блоков. Язык разметки должен четко обозначать, какая разметка допустима именно в данном документе, какая разметка обязательна, как отличить ее элементы от простого текста и что разметка значит.

SGML

SGML (Standard Generalized Markup Language) был официально принят в 1986 году в качестве международного стандарта (ISO 8879:1986) для описания независимых от устройств ввода/вывода и от вычислительной среды методов представления текстовой информации в электронной форме. Основой для его создания послужил довольно старый язык разметки GML(Generalized Markup Language), разработанный компанией IBM еще во времена первых персональных копьютеров. Если быть точным, то SGML - это метаязык, предназначенный для описания других языков

разметки.

Язык SGML - это типичное детище академической науки, изящная игрушка теоретиков. Его создание не было вызвано насущной практической необходимостью. Принципы, на которых строится этот язык, значительны и интересны; несомненно, идеология SGML оказала влияние на многие компьютерные разработки.

Однако сам по себе SGML не получил сколько-нибудь заметного распространения - до тех пор, пока в 1991 г. сотрудники Европейского института физики частиц (CERN), занятые созданием системы передачи гипертекстовой информации через Интернет, не выбрали SGML в качестве основы для нового языка разметки гипертекстовых документов. Этот язык - самое известное из приложений SGML - был назван HTML (HyperText Markup Language, "язык разметки гипертекста").

Цель появления SGML очень проста. В то время существовало несколько "языков разметки", ни один из которых не был совместим с несколькими платформами или даже программными пакетами. Появление SGML сделало возможным унификацию языков разметки, что было использовано для обеспечения гибкости и возможности передачи информации между приложениями и платформами.

SGML, в отличие от всех других языков разметки, созданных на его основе, использует принцип так называемой описательной разметки вместо процедурной. Подобная система использует элементы разметки, которые попросту предоставляют названия для отнесения отдельных частей документа к определенным категориям. Другими словами, тэги, такие как <para> или \end{list}, просто идентифицируют порцию документа и утверждают, что "эта часть является параграфом" или что "эта часть является концом начатого списка", и т.п. Система же, использующая процедурную разметку (сюда попадают текстовые процессоры, например, Microsoft Word) определяет, какая непосредственно обработка будет выполняться в конкретной точке текстового документа: "в этом месте вызвать такую-то процедуру с параметрами 5, e и z" или "передвинуть границу документа на 7 мм правее относительно какого-либо элемента, пропустить одну строку начать следующую с красной строки" и т.д. В SGML инструкции, которые необходимы для обработки документа с определенной конкретной целью (например, для форматирования), четко отделяются от описательной разметки, которая встречается внутри документа. Обычно они собраны вне документа в отдельных процедурах или программах.

При использовании описательной, а не процедурной разметки один и тот же документ может быть обработан разными программами, каждая из которых может применять свои собственные инструкции обработки к тем его частям, которые она считает важными. Например, программа анализа содержимого может полностью игнорировать сноски, тогда как программа форматирования может извлекать и собирать их для печати в конце каждой части. Различные виды инструкций обработки могут ассоциироваться с одной и той же частью файла. Например, одна программа может извлекать из документа фамилии людей и географические названия для создания индекса или базы данных, тогда как другая, обрабатывающая тот же самый текст, может печатать фамилии и названия отличающимся шрифтом.

SGML вводит также понятие типа документа, и, соответственно, способы его определения (document type definition, DTD). Документы считаются типизированными, так же как и другие обрабатываемые компьютерами объекты. Тип документа формально определяется его составными частями и их структурой. Скажем, можно определить тип документа таким образом, что он должен состоять из заголовка и, возможно, имени автора, за которыми следует аннотация и последовательность одного или более абзацев. Любой документ в отсутствие заголовка, в соответствии с этим формальным определением, не будет являться отчетом, так же как не будет им являться и последовательность абзацев, за которой следует аннотация, невзирая на то, насколько похож на отчет такой документ с точки зрения читателя-человека.

Поскольку документы относятся к известным типам, можно использовать специальную программу, называемую анализатором (parser), для того чтобы обработать документ, утверждающий, что он относится к конкретному типу, и проверить, действительно ли все элементы, требуемые для данного типа документов, присутствуют и находятся в правильной последовательности и корректно структурированы. Что еще более важно, разные документы одного типа могут обрабатываться унифицированным образом. Можно писать программы, использующие знания, заключенные в информационной структуре документа, которые, таким образом, могут быть более интеллектуальными.

SGML, как метаязык, позволяет определять конкретные языки (часто называемые "приложениями SGML"), ориентированные на конкретное применение. Пример тому - язык HTML, широко использующийся на WWW. Каждый такой язык описывается в виде DTD, определяя элементы и их атрибуты. Получив такой DTD, программное обеспечение для работы с SGML может корректно обрабатывать документы,

написанные в соответствии с этим DTD.

HTML

Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1991--1992 годах в стенах Европейского совета по ядерным исследованиям в Женеве (Швейцария). HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов (размечаемых «тегами»), служащих для создания относительно простых, но красиво оформленных документов. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже. Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащенностью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако, современное применение HTML очень далеко от его изначальной задачи. Например, тег <TABLE>, использованный для форматирования страницы, предназначен для создания в документах самых обычных таблиц, но, как можно убедиться, здесь нет ни одной таблицы.

Изначально HTML, как и положено SGML-приложению, разделял все особенности идеологии SGML. Из сорока с небольшим тегов HTML версии 1.2 (датированной июнем 1993 г.) всего три, да к тому же и не рекомендованных к использованию, тега осмеливались намекать на физические параметры представления документа.

Вся разметка была чисто логической, и лишь в описательной части стандарта, сопровождающей формальное определение тегов, можно было прочесть что-нибудь вроде "в графических броузерах действие этого тега может передаваться курсивным начертанием".

А первым (и долгое время единственным) графическим броузером в те далекие времена была программа Mosaic, разработанная, как и сам WWW, в научном учреждении - Национальном центре суперкомпьютерных приложений США (National Center for Supercomputer Applications - NCSA).

Так что нет ничего удивительного в том, что в этот "золотой век" никаких противоречий между официальными стандартами и их реализацией в броузерах еще не существовало. HTML неторопливо развивался, оставаясь в рамках парадигмы структурной разметки, и в апреле 1994 г. началась подготовка спецификации следующей версии языка - 2.0. Этим занимался образованный в том же году Консорциум W3 (W3 Consortium, сокращенно W3C), унаследовавший от CERN верховную власть и авторитет в мире WWW.

В настоящий момент консорциум, имеющий статус "международной некоммерческой организации", объединяет свыше 150 организаций-членов, в том числе фирмы Netscape, Microsoft и множество других. Однако в 1994-1995 гг. его членами были почти исключительно университеты и научные учреждения.

Столь "академический" состав W3C сказывался как на самих документах, публикуемых консорциумом, так и на процедуре (и особенно на сроках) их принятия. Достаточно сказать, что окончательный вариант HTML 2.0, единственным серьезным усовершенствованием в котором был механизм бланков (форм) для отсылки информации с компьютера пользователя на сервер, был окончательно утвержден лишь в сентябре 1995 г., когда в W3C уже полным ходом шло обсуждение HTML 3 (или, как его называли поначалу, "HTML+").

XML

XML начал вызывать огромный интерес и поддержку с того момента, как о нем было впервые заявлено в 1996 году. Он обеспечивает стандартный способ кодирования содержания, обеспечивая гибкий способ создания структур данных. В XML для разметки содержания на основе правил, составляемых разработчиком документа, используются теги. Этот набор правил называется описанием типа документа (Document Type Definition, DTD), и он позволяет разработчикам, применяющим XML, размечать различные документы. Сейчас документы всех этих типов можно создавать при помощи XML, для отображения содержания применяя HTML и Dynamic HTML (DHTML). XML также позволяет переназначать, переопределять и отображать содержание из одного источника при помощи других механизмов отображения. Например, хранить единую базу данных на сервере и поскольку сами данные отделены от информации об их представлении, одни и те же данные XML (будь то кулинарное меню или назначение врача) могут быть представлены различным образом на экранах компьютеров пользователей. Они могут быть отображены также на экране устройства, умещающегося в руке. Сам по себе XML-документ не указывает, будет ли, и каким образом, информация

отображена на экране. XML-документ содержит лишь данные. С помощью механизма таблиц стилей HTML отображает данные. Как Web-сервер, так и браузер управляют преобразованием XML-данных в формат HTML. К тому же данные XML могут обновляться автоматически, без обновления всей страницы в целом. Такое фрагментарное обновление XML делает HTML-страницы более эффективными и динамичными. ображать выделенные данные на нескольких разных устройствах.

Первый рабочий проект спецификации SGML был опубликован в ноябре 1996 года. Спустя некоторое время, в январе 1997 года появился первый анализатор XML, а в марте 1997 года начали выходить в свет первые приложения XML, и среди них — Lark. Затем, осенью, поддержка XML была реализована в браузере корпорации Microsoft. Наконец, в феврале 1998 года была опубликована рекомендация XML 1, и стала расти поддержка этого нового языка.

XML предоставит разработчикам инструменты, необходимые для выпуска новых видов приложений — передовых приложений, которые охватывают не только простые Web-приложения, но и базы данных, электронные коммерческие системы, а также фактически любые системы отображения информации. Это возможно, поскольку, в отличие от HTML, XML содержит все для работы с данными. Он направлен на структурирование данных, а внешнее представление данных возложено на HTML-ориентированные таблицы стилей.

XML является также многообещающим инструментом для intranet-технологий, поскольку позволяет разработчикам строить связи с базами данных, независимые от используемых систем. И что самое важное, средствами XML можно создавать различные структуры данных по заказу, для конкретных нужд отдельных отраслей. Эти структуры и базы данных можно просматривать при помощи всевозможных устройств отображения, не встраивая для этого специальные заказные интерфейсы, обеспечивающие просмотр одних и тех же данных на разных устройствах. В итоге, на сайтах больше не будет предупреждающих сообщений типа: "Для полноценного просмотра данных рекомендуется использовать браузер Netscape" (или Internet Explorer). Напротив, любой браузер, в котором предусмотрена поддержка XML, сможет обрабатывать и отображать данные, созданные при помощи XML.

Глава 2 Сравнение HTML и XML – что лучше

Возможности HTML

Язык HTML позволяет размечать в тексте:

1. Смысловую роль текстового блока (например: логическое ударение, заголовок (от первого до шестого уровня), параграф, пункт списка и др.), который обрабатывается браузером в соответствии со смыслом (например, в голосовых браузерах -- изменение интонации, в графических -- выделением курсивом, и т. п.) или настройками пользователя.

2. Гипертекстовые ссылки, которые значительно упрощают чтение множества связанных документов, ибо позволяют запросить документ с адресом, указанным в коде ссылки, простым щелчком мыши.

Эти управляющие коды используются для кодирования выделенных цветом либо подчеркиванием фрагментов текста или графических изображений, перехода с помощью выделенных слов к другому документу, другому текстовому блоку или рисунку. В качестве примера можно привести запись:

<A HREF="URL "> Фрагмент текста </A>

где «Фрагмент текста» - это часть документа, видимая пользователем в окне просмотра браузера, а URL - «место назначения» гипертекстовой связи. Адрес этого «места» может быть абсолютным - с заданием полного имени сервера и именем файла документа назначения; относительным, при котором предполагается, что имя сервера и начальный каталог те же, что и у документа, содержащего гипертекстовую ссылку. В HTML реализована поддержка механизма специальных гипертекстовых ссылок, которые обеспечивают связь данной публикации с другими публикациями. Гипертекстовая ссылка - это адрес другого HTML-документа или информационного ресурса Internet, который тематически, логически или каким-либо другим способом связан с публикацией, в которой эта ссылка определена. Ссылка состоит из двух частей. Первая их них - это то, что визуализируется в поле WEB-страницы. Она называется «указатель ссылки» (anchor). Вторая часть, дающая инструкцию браузеру, называется адресной частью ссылки (Universe Resource Locator или URL-адрес).

Пример:

<A HREF = "http://polyn.net.kiae.su/ altai/index.html">

Здесь тег - контейнер <A> (anchor), использует атрибут HREF, обозначающий гипертекстовую ссылку, для записи этой ссылки в форме URL. Указатель может

быть как относительным, так и абсолютным. Данная ссылка указывает на документ с именем "index.html" в каталоге "altai" на сервере "polyn.net.kiae.su", доступ к которому осуществляется по протоколу "http". Возможно использование локального адреса в том случае, если файл находится на ПЭВМ, где запущена программа просмотра WWW, а не на сервере WWW. Между кодами <A> и </A> можно поместить текст любого объема, код <IMG> для вставки графики или сочетание того и другого.

Кроме ссылок на другие документы часто используются ссылки на разные части текущего документа. Например, большой документ читается лучше, если он имеет оглавление со ссылками на соответствующие разделы. Для построения внутренней ссылки сначала нужно создать указатель, определяющий место назначения. Например, сделать ссылку на текст определенной главы документа. Сначала размещается указатель и присваивается имя при помощи параметра name тега <a>. При этом параметр href не используется и браузер не выделяет содержимое тега <a>. Например:

Теперь создается сама ссылка. Для этого в параметре href указывается имя ссылки с префиксом #, свидетельствующим о том, что это внутренняя ссылка.

Глава 5

Если пользователь щелкнет кнопкой мыши на словах «Глава 5», браузер выведет соответствующую часть документа в окно просмотра.

При использовании гипертекстовых связей программа просмотра извлекает документ, форматирует его и затем показывает начиная с верхней части окна просмотра. Однако код связи позволяет также немедленно "прокрутить" документ до метки в документе назначения. Код гипертекстовой связи такого рода имеет вид:

<A HREF="URL #метка"> Фрагмент текста </A>

Метка должна находиться в HTML-документе назначения:

<<A NAME="метка"> текст, видимый пользователю </A>

Гипертекстовые связи могут изменять положение просматриваемого документа на экране ПЭВМ:

<<A HREF="#метка"> Текст <A/>

Гипертекстовые связи такого типа позволяют начинать просмотр документа с его общего плана (упорядоченного или неупорядоченного списка), где каждый элемент представляет собой код гипертекстовой связи, адресующий к соответствующему разделу документа.

3. Гарнитуру, кегль, начертание, цвет шрифта для визуального вывода. (форматирование текста)

Теги управления абзацами

<P ALIGN=CENTER/LEFT/RIGHT >...</P> - тег нового абзаца, используется в формате одиночного тега или контейнера. При использовании в форме одиночного тега концом абзаца считается начало следующего т.е следующий тег <P>. Атрибут ALIGN задает выравнивание элементов абзаца, значение по умолчанию LEFT

<P>...</P> или <P>	Этот абзац выравнивается по левому краю. И этот абзац тоже.
<P ALIGN=CENTER>	Этот абзац выравнивается по центру.
<P ALIGN=RIGHT>	Этот абзац выравнивается по правому краю.

Теги управления переносом

,<NOBR>...</NOBR>, <WBR> - теги управления разрывами и переносом строк в тексте документа. При разрыве строки межстрочный интервал не увеличивается.

 	Используется для указания места принудительного разрыва. Пример: <P>ФИО: Иванов С.С.</P> Будет выглядеть так: ФИО:Иванов С.С.

<NOBR>...</NOBR>	Используется для запрета разрыва текста, помещенного в данный контейнер. Пример: <NOBR>Это лучше не разрывать</NOBR> при необходимости переноситься на новую строку целиком, а не так: Это лучшене разрывать
<WBR>	Используется для указания рекомендуемого места для разрыва строки. Может быть вложенным в контейнер <NOBR>...</NOBR>. Пример: <NOBR>42301<WBR>810600000000001</NOBR> - номер счета заданный таким образом при помещении в поле уже своей ширины, разорвется после балансового счета: 42301810600000000001

Теги выделения структуры документа

<H1>...</H1>, ... ,<H6>...</H6> - контейнерные теги шестиуровневых заголовков документа. Имеют атрибут ALIGN (*по умолчанию LEFT*) для выравнивания заголовка.

<H1>...</H1>	Заголовок 1 уровня
<H2>...</H2>	*Заголовок 2 уровня*
<H3>...</H3>	Заголовок 3 уровня
<H4 ALIGN=LEFT>...</H4>	Заголовок 4 уровня по левому краю
<H5 ALIGN=CENTER >...</H5>	*Заголовок 5 уровня по центу*

<H6 ALIGN=RIGHT>...</H6>	Заголовок 6 уровня по правому краю

Теги смыслового выделения текста

Контейнеры для смыслового выделения заключенного в них текста на Web-страницах. Способ выделения зависит от типа используемого броузера, но главное назначение этих тегов передача читателям логики автора.

<CODE>...</CODE>	Компьютерный код - Function Sum(a,b);
<CITE>...</CITE>	Выделение цитат - *Цитата*
<KBD>...</KBD>	Клавиатурный шрифт - Клавиатура
<SAMP>...</SAMP>	Выделение примеров -Пример
<STRONG>...</STRONG>	Выделение важных фрагментов - Важно
<VAR>...</VAR>	Выделение имен (*i, j, k*) переменных
<DFN>...</DFN>	Выделение определений - *Определение*
<EM>...</EM>	Расставить акценты - *Акцент*
<BLOCKQUOTE>...</BLOCKQUOTE>	Выделение фрагмента текста в большом блоке текстовом блоке на странице. Вот фрагмент который мы хотели выделить из текстового блока в документе. Таким образом выделенные фрагменты текста отображаются броузером.

Теги стилистического выделения текста

Данная группа контейнерных тегов применяется для стилистического выделения элементов текста. Допускается любая комбинация нижеперечисленных тегов.

<B>...</B>	Выделение полужирным шрифтом
<I>...</I>	*Выделение курсивом*
<TT>...</TT>	Выделение телетайпным шрифтом
<U>...</U>	Выделение подчеркиванием
<STRIKE>...</STRIKE>	~~Выделение перечеркиванием~~
^{...}	Шрифт в верхнем индексе
_{...}	Шрифт в нижнем индексе
<SMALL>...</SMALL>	Мелкий шрифт
<BIG>...</BIG>	Крупный шрифт

Дополнительные теги форматирования

<HR> - тег вставки линии-разделителя. Применяется для визуального разделения текста, при помощи горизонтальных линий (*не путайте с графическими изображениями в форме разделителей*). При отображении линии-разделителя в документе, до и после нее, броузер добавляет разделение абзацев. Формат линии-разделителя задается при помощи следующих атрибутов:

· ALIGN - выравнивание (*LEFT / RIGHT / CENTER*);

· WIDTH - ширина линии (*пикселы или проценты к ширине окна WIDTH=50%*);

· SIZE - высота линии (*пикселы*);

· COLOR - цвет линии;

· NOSHADE - отключить эффекты 3-х мерности;

Использование шрифтов в документах

При использовании различных шрифтов для оформления текста следует помнить, что у пользователя может не оказаться шрифта, использованного вами для создания документа. Если вы используете редкие или нестандартные шрифты, то браузер пользователя может не подобрать шрифт для корректного отображения документа.

Для определения шрифта текста в HTML документах применяется контейнер <FONT>...</FONT> и одиночный тег <BASEFONT>.

Тег <BASEFONT> задает базовые параметры шрифта, общие для всего документа. Действие базовых установок может быть отменено атрибутами нового тега <BASEFONT>.

Контейнер <FONT> применяется для изменения параметров шрифта отдельных элементов документа, которые необходимо отобразить шрифтом отличным от базового. Действие его атрибутов ограничивается фрагментом документа, заключенным в данный контейнер, и он может быть вложенным по отношению к другим тегам форматирования текста.

Для задания характеристик шрифта в тегах <FONT>...</FONT> и <BASEFONT> используются следующие атрибуты:

· FACE - Задает имя шрифта (*или перечня шрифтов - по мере убывания предпочтения*) на компьютере пользователя. В случае отсутствия текст отображается шрифтом, заданным по умолчанию в броузере пользователя. Например:

<FONT FACE="Arial">Пример Arial</FONT> - Пример Arial

· SIZE - абсолютный или относительный размер шрифта. Относительный размер это размер шрифта относительно стиля Normal (*SIZE=3*) или размера заданного тегом <BASEFONT>. Минимальное абсолютное значение размера шрифта 1, максимальное 7. Например:

<FONT SIZE=4>4 абсолютный шрифт</FONT> - 4 абсолютный шрифт

<FONT SIZE=+1>4 относительный шрифт</FONT> - 4 относительный шрифт

· COLOR - цвет шрифта. Например:

<FONT COLOR=RED>Красный шрифт</FONT> - Красный шрифт

<FONT COLOR=#FF0000>Красный шрифт</FONT> - Красный шрифт

Полученные в данном разделе навыки, по форматированию текста, закрепим конкретным примером:

4. Специальные символы (выходящие за рамки ASCII символы пунктуации, математические символы, греческие и готические буквы, стрелки и т. п.)

5. Формы для введения пользователем данных, которые позднее подвергаются обработке. Формы и другую информацию можно обрабатывать с помощью специальных серверных программ (например, на языках PHP или Perl).

6. Открытие мультимедийных файлов, выводимых как непосредственно браузером (например, изображения в форматах JPEG, GIF или PNG; аудиофайлы MIDI и др.), так и внешними приложениями, «встраиваемыми» в окно браузера (Flash-анимация, Java-апплеты и прочее).

Использование графики в документах позволяет повысить привлекательность ваших Web-страниц, делает изложенный материал более доступным для восприятия, а в некоторых случаях (*искусство, реклама*) без нее просто не обойтись.

Web-броузеры поддерживают множество графических форматов, но наиболее часто используются GIF и JPEG (некоторые форматы требуют установки дополнительных программных компонентов броузера).

Вставка изображений в документ

Для вставки изображения в документ используется одиночный тег <IMG>. Местоположение изображения на странице и его выравнивание относительно текста задается следующими атрибутами:

· SRC - URL изображения;

· ALIGN - выравнивание текста относительно изображения (режимы с обтеканием текста: LEFT - изображение слева, текст обтекает справа / RIGHT-изображение справа, текст обтекает слева; режимы без обтекания текстом: TOP - по верхнему краю изображения / MIDDLE - по центру изображения / BOTTOM - по нижнему краю);

· WIDTH - ширина изображения (*пикселы*);

· HEIGHT - высота изображения (*пикселы*);

· ALT - текстовое описание-альтернатива, для тех кто отключил загрузку изображений;

· BORDER - ширина рамки (*по умолчанию BORDER=1*);

· HSPACE - пустое поле от изображения по горизонтали;

· VSPACE - пустое поле от изображения по вертикали;

· ISMAP - признак карты-ссылок (*обработка сервером*);

· USEMAP - признак карты-ссылок (*обработка клиентом*);

Примеры тега <IMG>:

<IMG SRC="pic1.gif" ALIGN=MIDDLE>

<IMG SRC="pic2.jpg" HSPACE=20 VSPACE=20 ALT="Здесь изображение офиса нашей компании">

<IMG SRC="pic3.jpg" WIDTH=120 HEIGHT=160 ALIGN=LEFT BORDER=5>

Возможности XML:

1. Лучший контроль над размещением информации.

2. Меньшая загрузка Web-сервера благодаря возможностям по доступу к информации на клиентской стороне. (независимость от сервера)

3. Применение различных типов гиперссылок (hyperlinks).

4. Возможность распространения различных видов информации в Internet и intranet.

5. Меньшее количество проблем, возникающих при отображении больших страниц (long pages).

Лучший контроль над размещением информации

В XML информация о компоновке располагается отдельно от непосредственного содержания, таким образом, когда дизайнер примет решение изменить компоновку сайта, он просто вносит изменения в используемую таблицу стилей. Содержание при этом остается неизменным. В этом заключается главное отличие от концепции HTML, что позволяет различным механизмам использовать значительно более гибкий формат обмена информации. Таблицы стилей могут применяться для форматирования содержания документов в различных приложениях.

XML позволяет сопоставлять стили с конкретными структурными элементами. Это означает, что разработчик может быстро применить стилевой формат для определения структурных элементов, например, сущностей, содержащих объявления изображений, особенных форматов абзацев, и даже стилей для различных типов механизмов связывания (linking mechanisms).

Независимость от сервера

Одной из наиболее важных функциональных возможностей XML является то, что документы не нуждаются в жесткой привязке к серверам. Используя так называемую объектную модель документа (Document Object Model, DOM) можно создавать XML-документы, в которых отображаются либо все данные, либо лишь часть этих данных. Предположим, что вы создаете XML документ -- простую адресную книгу. При помощи HTML вы могли создать форму, позволяющую находить имя в адресной книге. Разумеется это бы потребовало пересылки поискового запроса из формы к серверу всякий раз, когда пользователю необходимо обратиться к тому или иному им ни в адресной книге.

Благодаря применению DOM в XML в документе может содержаться весь полный список, и если подключена таблица стилей, то из этого списка будет отображаться лишь затребованная информация. Все другие элементы документа могут быть при этом скрыты. Если же пользователю нужна более полная информация, то вместо того, чтобы посылать запрос к серверу, понадобится лишь отобразить остальную часть скрытой информации с помощью скрипта, выполняемого браузером. Такой механизм таблиц стилей позволяет работать с адресной книгой как в Internet, так и вне его.

Лучший контроль над большими документами

Вы когда-нибудь пытались просмотреть большую Web-страницу при помощи современных браузеров? HTML не позволяет отметить и выбрать с отдельный раздел для просмотра.

XML позволяет просто решить эту проблему, поскольку все XML-документы структурированы и правильно оформлены. В XML вам не придется "срезать острые углы", как в HTML, применяя различные элементы разделов, вы можете разбить единый документ на разделы. Такая многоуровневая структура напоминает иерархическое представление папок (folders) в Windows Explorer. Из этого следует, что XML обеспечивает возможность поиска по всему документу, не прибегая к созданию отдельных скриптов.

Как выглядит XML-документ

XML-документ очень похож на SGML- или HTML-документ. Как вы уже знаете, в него входят элементы, атрибуты и сущности, а также комментарии и другой стандартный текст. Но существуют некоторые особенности, которые отличают XML-документы от младших и старших родственников по линии языков разметки. Сначала рассмотрим простой XML-документ:

```
<?XML version="1.0"?>
<!DOCTYPE MEMO SYSTEM "http://www.site.com/dtds/memo.dtd">
<MEMO>
<headER>
<TO>
TO:
<NAME>
John Doe
</NAME>
<CC/>
</TO>
<FROM>
From:
</FROM>
<SENDER>
Betty North
</SENDER>
```

</headER
<!CThis is the start of the memo text - ->
<MEMOTEXT>
Please take note our phone number has changed.
</MEMOTEXT>
</MEMO>

Во-первых, обратите внимание, что данный документ начинается с исполняемой инструкции <?XML version="1.0"?>. Эта строка кода называется объявлением разметки XML, она указывает процессору, что данный документ является XML-документом, в котором для структурирования применяется XML версии 1.0. Хотя это объявление не является обязательным, вам следует его включать в текст кода всегда, для того чтобы процессор "понимал", что он имеет дело с XML-документом. Иначе процессоры или браузеры могут рассматривать его как стандартный HTML-документ и неправильно обработать содержащуюся в нем информацию.

Во-вторых, обратите внимание, что, в отличие от SGML, присутствие DTD не является обязательным, однако в данном случае мы включили одно такое объявление. Это DTD хранится на сервере по адресу www.site.com/dtds в файле Memo.dtd. Оно определяет все элементы и сущности, используемые в данном документе.

Обратите также внимание на элементы и применение в них открывающих и закрывающих тегов. Они выглядят точно так же, как элементы, применяемые в HTML. Однако в XML требуется обязательное наличие как открывающих, так и закрывающих тегов. Теоретически, в HTML вы можете опустить закрывающий тег, и браузеры по-прежнему будут правильно воспроизводить данный документ. В XML такой документ не будет правильно обработан.

Теперь рассмотрим элемент </CC>. В отличие от элементов, которые позволяют открывать и закрывать теги, пустые элементы, такие как <CC/>, немного модифицированы и не обязательно должны сопровождаться открывающими тегами где-либо в содержании текста. Напротив, пустые элементы следует представлять как маркеры, на месте которых может появиться что-либо, или может быть определено значение по умолчанию. Пустые элементы всегда обозначаются при помощи имени, элемента, за которым следует наклонная черта, например, <LATESTIME/>, но для них всегда могут быть установлены границы с помощью начального и оконечного тегов, причем между тегами нет никаких значений — <LATESTTIME><LATESTIME/>. Как видите, XML-документы состоят из

содержания и разметки.

Также обратите внимание на то, как один элемент может быть встроен в другие элементы, например, <MEMO></MEMO>, <head></head> и <MEMOTEXT></MEMOTEXT> — точно так же, как они могли быть встроены друг в друга в HTML. Вы можете помещать комментарии (как и в HTML) — подобно тому, как это сделано перед самим текстом служебной записки (memo). Обратите внимание также, что отсутствуют форматирующие теги. Вместо этого вся работа по форматированию выполняется при помощи отдельной таблицы стилей, определенной в отдельном документе.

Применение множественных гиперссылок

XML предоставляет стандартную модель связывания, основанную на собственной спецификации -- расширяемом языке связывания (Extensible Linking Language, Xlink). В HTML в качестве указателя связи, или URL, могут применяться лишь символьные типы данных. Сущности не применяются. В URL не могут содержаться условные обозначения (notations), разделяющие различные типы данных. А внутренние связи, применяемые для связи с другими ссылками в том же самом документе, используют атрибут NAME типа элемента, например, <A NAME="bottom">. Это относительно простое связывание.

Связывание в XML является, однако, более сложным, чем в HTML. Благодаря XLL (extensible Link Language) -- языку описания связей для внедрения других документов XML и двунаправленных ссылок -- XML предлагает развитые механизмы связывания, которые:

1. Предоставляют управление семантикой связи.

2. Применяют спецификацию расширенных связей (extended links), благодаря которой обеспечивается более двух связей.

3. Поддерживают указатели на внешние ресурсы, благодаря применению спецификации расширенных указателей (extended pointers, Xpointers)

Не прибегая к профессиональной терминологии, можно сказать, что опции расширенного связывания обеспечивают документы XML:

1. двухсторонними связями;

2. внешним управлением связями (то есть такими связями, которыми можно управлять, находясь за пределами содержания данного документа);

3. связями, которые обеспечивают доступ к кольцу сайтов (ring of sites) или позволяют открывать множество окон;

4. связями с различными источниками;

5. атрибутами связей.

Типы документов XML

В предыдущем примере было продемонстрировано несколько возможностей. Эти возможности охватывают шесть типов объектов разметки, используемых в XML:

- Элементы (elements)
- Ссылки на сущности (entity references)
- Комментарии (comments)
- Исполняемые инструкции (processing instructions)
- Отмеченные разделы (marked sections)
- Типы документов (document types)

В листинге, приведенном выше, присутствуют элементы, комментарии, отмеченные разделы и типы документов. Последующие примеры, приводимые в этой работе, показывают, как использовать эти и другие объекты (исполняемые инструкции и ссылки на сущности) в конкретных случаях.

Хотя мы включили DTD в листинг, это не было необходимостью. Именно поэтому XML хорошо подходит для работы в Web — не нужно знать DTD или же структурную модель, применявшуюся при создании XML-документа. Это не означает, что XML не работает с DTD. Это лишь означает, что если в документе не используется описание типа документа (DTD), то эти описания должны быть встроены в сам документ. Именно эта возможность — либо использовать DTD, либо включать форму структуры документа — и делает XML столь подходящим инструментом для работы в Internet- и intranet-приложениях.

Документы, у которых есть описания типов документов (DTD), которым они подчиняются, называются действительными (valid). Документы, у которых нет DTD, но они все же соответствуют стандарту XML, называются правильными (well-formed). Правильные документы могут не содержать DTD, однако они должны отвечать некоторым простым правилам, для того чтобы с ними можно было

использовать таблицы стилей и механизмы связывания. Рассмотрим в общих чертах каждый тип документа.

Правильные документы

Что же представляет собой правильный документ? Правильным является такой XML-документ, который соответствует синтаксису XML, используемому в данном документе. Например, если вы не включите оконечный тег при расстановке элементов в документе, если вы забудете включить объявление XML-документа, которое должно размещаться в самом его начале, или же если документ содержит символы, которые не могут быть обработаны анализатором, или же являются недействительными (invalid), то такой документ не будет считаться правильным. Некоторые конструкции разметки разрешено размещать в строго определенных местах. Поместите эти конструкции в любое неверное место — и ваш документ не будет правильным.

Примечание
Правильный XML-документ является также правильным SGML-документом.

Но правильные XML-документы — это нечто большее, чем просто документы, которые соответствуют синтаксису XML. Кроме того, эти документы должны отвечать следующим условиям:

- В начальном теге атрибут не должен встречаться более одного раза.
- Нельзя включать ссылки на внешние сущности в строковый атрибут.
- Необходимо объявлять все сущности, кроме тех, которые используются как часть языка XML.
- Нельзя ссылаться на двоичную сущность, размещенную непосредственно в содержании документа. Двоичные сущности могут фигурировать лишь в атрибутах ENTITY.
- Нельзя создать рекурсивный параметр (recursive parameter) или же рекурсивную текстовую сущность — ни прямым, ни косвенным способом.
- Параметрические сущности должны быть объявлены прежде, чем вы начнете их использовать в документе.
- Любые непустые теги должны быть правильно вложены (nested).
- Имя, указанное в оконечном теге элемента, должно соответствовать типу элемента в начальном теге.
- Нельзя помещать угловые скобки (<) в замещаемый текст какой бы то ни было сущности.

Действительные документы

Допустим, у вас есть правильный XML-документ. Вы позаботились о выполнении всех ограничений по разметке, "расставили все точки над i", и оформили документ в соответствии с синтаксисом XML. Но правильный документ не является действительным до тех пор, пока в нем нет объявления типа документа (DTD). Причем, данный документ должен отвечать ограничениям, налагаемым данным объявлением.

Каждый действительный документ должен начинаться с информации заголовка содержащей следующее:

- Описание правил структурирования (structural rules), которым должен отвечать данный документ.
- Список любых внешних ресурсов или внешних сущностей, составляющих какую-либо часть данного документа.
- Любые объявления внутренних ресурсов или внутренних сущностей.
- Любые условные обозначения или же ресурсы, представленные в форматах, отличных от XML. Эти обозначения и ресурсы должны определять требуемые вспомогательные приложения (helper applications),
- Списки ресурсов, представленных в форматах, отличных от XML (например, двоичных сущностей), имеющихся в документе.

Сравнение XML и HTML

И HTML, и XML документы состоят из элементов, каждый из которых включает «начальный тэг» (<order>), «конечный тэг» (</order>) и информацию, заключенную между этими двумя тэгами (которая называется содержимым элемента). Элементы могут быть аннотированы атрибутами, содержащими метаданные об элементе и его содержимом.

Однако между HTML и XML есть существенные отличия. XML чувствителен к регистру, в то время как HTML — нет. Это значит, что в XML начальные тэги <Table> и <table> различны, тогда как в HTML — это одно и то же. Другое различие между HTML и XML в том, что XML представляет концепцию правильного построения. Правила построения XML устраняют некоторую неопределенность, присущую обработке таких языков разметки как HTML, вводя такие постулаты, как то, что все значения атрибутов должны быть заключены в кавычки, и что у всех элементов должны быть или начальный и конечный тэги, или явное указание того, что это пустые элементы. Краткое описание правил построения дается в XML FAQ.

Самое существенное различие между HTML и XML в том, что в HTML есть предопределенные элементы и атрибуты, поведение которых предопределено, в то время как в XML такого нет. Вместо этого, авторы документа могут создавать собственные XML-словари, предназначенные именно для их приложения или деловых нужд. В настоящее время существуют XML-словари для огромного количества отраслей промышленности и приложений: от финансовых картотек (XBRL) и финансовых операций (FpML) до Web-документов (XHTML) и сетевых протоколов (SOAP). Отсутствие предопределенных элементов и атрибутов, которые определяют, как формируется или отображается XML-документ, дает возможность авторам сосредоточиться на создании документов, которые содержат только существенную семантическую информацию в их конкретной предметной области. Отделение содержимого от представления, ставшее возможным с XML-словарями, существенно увеличивает возможности повторного использования информации и перенацеливания содержимого.

Заключение

Невзирая на тот факт, что HTML и XML созданы с использованием одной технологии, а также что XML применяет значения, атрибуты, тэги для того чтобы форматировать структурированные документы, XML обладает довольно значительными отличиями.

HTML считается простым языком для образования разметки страниц сайта. Он имеет фиксированное число тэгов и расплывчатый перечень правил. В HTML нужно закрывать некоторые тэги, прочие нельзя закрывать. А часть можно и не закрывать и закрывать. Безусловно, это было основополагающей идеей сотворения HTML.

Идея была великолепна на первоначальных стадиях формирования Web, когда веб-ресурсы состояли из простого содержимого, большего и не было нужно. Впрочем, в сегодняшних сайтах, где странички зачастую формируются посредством средств публикации, а информационная составляющая отправляется из БД на устройство беспроводного доступа и принтер или Web-страницу, скудность возможностей HTML стает причиной нарушения круговорота данных.

HTML, помимо того, считается только языком форматирования. HTML не включает данных относительно контента, что снова-таки ограничивает доступность данных в разных областях. И, безусловно, HTML был задуман лишь для применения в веб.

Разметка, которая основана на XML, в свой черед, имеет устойчивые правила и может использоваться далеко за границами Сети — применяется в агрегаторах новостей, сервисе XML лимитов Яндекс, картах сайта (Sitemap) и многое другое. При разметке веб-документа документ заключен в тэги, которые распознает любое окружение с поддержкой XML.

Список используемых источников

1. Женнет Ж. Палимпсесты: Литература второго уровня, М., 1982.
2. Nelson T. Literary machines. Sausalito, CA: Mindful Press, 1993.
3. Руднев В.П. Словарь культуры XX века. — М.: Аграф, 1991.
4. Субботин М.М. Итоги науки и техники. Сер. Информатика. Т. 18. М.: ВИНИТИ.
5. Conklin J. Hypertext: an introduction and survey//Computer, 1981. Vol. 20. № 9.
6. Овчинников В.Г. Автоматизированные ГТС: назначение, архитектура и перспективы развития // Научно-техническая информация. Сер. 1. 1990. № 12.
7. Эпштейн В.Л. Введение в гипертекст и гипертекстовые системы. [Электронный документ]. http://www.ipu.rssi.ru/publ/epstn.ht). Проверено 12.02.2006 г.
8. Бутов М. Отчуждение славой // Новый мир. — 2000. — № 2. — С. 188–209.
9. http://spintongues.msk.ru/calvino.html
10. Барт, Р. Избр. работы: Семиотика. Поэтика. — М.: Прогресс-Универс, 1994. С. 14.