



Содержание

Об авторах	14
Предисловие переводчика	17
Введение в XML	19
Глава 1. Складываем мозаику XML	40
Биты и части	40
Определение типа документа	41
Правильные и состоятельные документы	43
Таблицы стилей	44
Расширяемый язык таблиц стилей XSL	50
Анализаторы	52
Создание ссылок в XML	55
Комментарии в XML	57
Новые птенцы в нашем гнезде	57
Пространства имен XML	58
XML-схемы	59
Просмотр XML-файлов	61
XML в реальном мире	65
Формат определения канала	65
Химический язык разметки	65
Открытый финансовый обмен	66
Заключение	67
Глава 2. Правильные и состоятельные документы	69
Правильные документы	69
Приступаем к созданию документа	70
Элементы	71
Атрибуты	75
Компоненты	76
Отложенный разбор участков данных	86
DTD: состоятельный документ	88
Объявление XML	89
Описание типа документа	90

Определение типа документа	91
Описание элементов	92
Описание списка атрибутов	101
Описание компонентов	112
Команды приложений	118
Условные разделы	121
Стиль описания	122
Заключение	124
Глава 3. XML-схемы	125
Определение типа документа XML как схема	127
XML-схема: общие вопросы	128
Трудности написания хорошего DTD	128
Нерасширяемость DTD	128
DTD плохо описывает данные XML	128
DTD не поддерживает пространства имен	130
Ограничения описательной способности DTD	130
DTD и содержание элемента по умолчанию	130
Проверка определения типа документа	130
Создание базы данных при помощи XML	132
XML-данные: предлагаемое решение	135
Простейший пример XML-данных	135
Более сложный пример XML-данных	138
Свойства XML-данных	147
Типы данных	150
Описание содержания документа	151
Описание содержания документа – начнем с простого	153
Узлы DCD и типы ресурсов	155
Как элементы и атрибуты рассматриваются в DCD	155
Заключение	162
Глава 4. Пространства имен	163
О чем говорится в этой главе	164
Что такое пространство имен	164
Идентификация и описание пространств имен	166
Синтаксис пространств имен	166
Описание пространств имен	166
Пространства имен и область действия	168
Атрибуты и пространства имен	169
Вывод элементов из области действия	170
Зачем нужны пространства имен	171
Уникальное определение элементов и атрибутов	171
Повторное использование схем	171
Обучение агента пользователя	172

Чего не может пространство имен	173
Ожидаемое поведение агента пользователя	174
Применение пространств имен	174
Таблицы стилей в Internet Explorer 5	174
Расширяемый язык таблиц стилей	176
Формат описания ресурсов RDF	178
Заключение	180
Глава 5. Ссылки и указатели в XML	181
Формирование ссылок в HTML	182
Простые ссылки	182
Немного терминологии	182
Различие между связыванием и адресацией	183
Указатели в HTML	183
Простые ссылки в XML	185
Определение тэгов ссылки	185
Атрибуты, предлагаемые спецификацией XLink	187
Атрибут xml:attribute	189
Совместимые с XLink агенты пользователя	190
Обзор терминологии	190
Расширенные ссылки	193
Встроенные расширенные ссылки	194
Внешние расширенные ссылки	196
Использование внешних расширенных ссылок	198
Малая сеть intranet	198
Большая сеть intranet	199
Поведение агента пользователя	200
Дистанционное комментирование документов	201
Обслуживание ссылок	202
X-указатели	203
Синтаксис локатора	205
Синтаксис X-указателей	207
Абсолютное место	208
Относительное указание места	209
Ключевые слова относительного указания	210
Использование ключевых слов	211
Указание с помощью атрибута	213
Интервальный терм места	214
Строковый терм места	214
X-указатели и определение типа документа	215
Заключение	216
Глава 6. Объектная модель документа XML	217
О чем говорится в этой главе	217
Общее представление о моделях документа	218

Дерево XML-документа	220
Документ XML как совокупность объектов	221
Объекты XML	222
Возможные свойства	222
Типы узловых объектов	223
Интерфейс приложения для объектной модели документа	223
Значение общепринятого интерфейса приложения	224
Язык определения интерфейсов группы управления объектами	224
Статус объектной модели документа	226
Интерфейсы объектной модели документа	227
XML в браузере IE5	233
Островок XML	233
Элемент ActiveX для XML	234
Примеры интерфейсов объектной модели документа	235
Интерфейсы Document и Node	235
Интерфейс Node	236
Интерфейс Document	241
Методы интерфейсов Node и Document	246
Интерфейс CharacterData	248
Интерфейс Attr	250
Интерфейс Element	251
Интерфейс узла Text	255
Интерфейс Comment	255
Интерфейс Processing Instruction	255
Интерфейс DocumentType	256
Интерфейс Notation	257
Интерфейс Entity	257
Интерфейс EntityReference	257
Некоторые простые реализации	258
Основной рекурсивный цикл	258
Простое оформление стилями	260
Простые таблицы	262
Подготовка слайдов	264
Другие примеры	273
XML и поисковые машины	274
Заключение	274

Глава 7. Просмотр XML-документов	275
HTML в сравнении с XML	275
Таблицы стилей	276
Потоковые объекты	277

Просмотр в браузере	278
Способы демонстрации XML-файлов	279
Демонстрация на различных устройствах	279
Демонстрация приложениями пользователя	279
Каскадные таблицы стилей	280
Что такое каскадная таблица стилей CSS	280
Простое правило стиля CSS	280
Соединение таблицы стилей и документа	282
Правило стиля	283
Свойства и значения	285
Формы правил каскадных таблиц стилей	287
Каскадирование и наследование	290
Рамки	291
Классы	294
Преобразование XML-документов	299
Преобразование вручную	299
Использование анализатора XMLparse.exe	300
Преобразование XML со «старой» таблицей стилей XSL	305
Язык Spice	305
Концепции языка Spice	306
Потоковые объекты языка Spice	308
Режимы и непоследовательное воспроизведение	310
Таблицы стилей, зависящие от системы воспроизведения	312
Графика	313
Присоединение таблиц стилей Spice	313
Уровень разработанности языка Spice	314
Заключение	314
Глава 8. Расширяемый язык таблиц стилей XSL	315
О чем говорится в этой главе	316
Краткий обзор	316
Здравствуй, XSL!	317
Потоковые объекты	318
Что представляют собой шаблоны XSL	318
Построение дерева XSL	320
Построение результирующего дерева из исходного	324
Пространства имен и таблицы стилей XSL	327
Атрибуты элемента xsl:stylesheet	329
Правила шаблона таблиц стилей XSL	330
Разрешение конфликтов сопоставлений	335
Форматирующие объекты, задающие размещение	336
Простые форматирующие объекты	336

Потоковые объекты содержимого	337
Применение стилей	340
Преобразование CSS в XSL	341
Простая обработка	341
Утраченные форматирующие объекты	342
Сложное применение стилей	343
Обработка пробельных литер	344
Пространство имен CSS	344
Будущее языка XSL	346
Заключение	346
Глава 9. XML и уровни данных	348
Методы доставки XML-документов	349
Электронный список телефонов	350
Создание XML на SQL-сервере	351
Использование SQL Server Web Assistant	354
Создание XML-данных в промежуточных системах	367
HTML-форма для обновления списка телефонов	375
Заключение	382
Глава 10. XML на стороне сервера	383
Причины использования XML на сервере	383
Клиенты: агенты, браузеры и другие	385
Система хранения технических статей	385
Клиент	386
Сервер	388
Публикация статей	392
Рассмотрение архитектуры ядра	393
Компромиссы в системе клиент-сервер	394
Вопросы передачи данных	395
Создание XML-файла на стороне клиента	397
Пользовательский интерфейс	397
Оформление параметров поиска	398
Обработка XML-документа, возвращенного сервером	400
Управление XML в Active Server Pages	401
Глобальные объекты сервера	402
Загрузка XML-строки	402
Получение корня дерева разбора	403
Подготовка ресурсов базы данных	403
Обход дерева разбора	404
Извлечение параметров	405
Получение ответов на запросы пользователя	407

Как реагирует клиент	410
Подготовка XML-файла к разбору на стороне клиента	410
Подготовка к работе с результатами	412
Заполнение таблицы	414
Представление материалов	416
ASP для презентации	417
Как работает XSL-процессор	418
Пространства имен, метаданные и будущие приложения	422
Пространство имен XML	422
XML-данные	423
Заключение	424
 Глава 11. Учебный пример «Туристический маклер»	425
Приложение «Туристический маклер»	426
Решение	426
Архитектура	427
Трехуровневая архитектура	428
Трехуровневая архитектура, использующая XML	428
Службы данных приложения	432
Базы данных	432
Определения типа документа для XML	435
Реализация при помощи ASP и ADO	439
Что делать дальше	446
Бизнес-службы	446
Пример	447
Реализация	448
Службы пользователя	459
Пример	459
Форматирование у клиента при помощи CSS	460
Форматирование на сервере при помощи XSL	460
Реализация	463
Заключение	465
Ссылки для получения дальнейшей информации	466
 Глава 12. «Сорняки Эль Лимона»: заказная издательская	
Web-система на основе XML	467
Как мы попали в этот переплет	467
Почему не годились простые Web-страницы	468
Почему я выбрал XML	468
Почему я выбрал статические Web-страницы	470
Почему я выбрал Java	471
Создание XML-документа	472

Пример документа	472
Определение типа документа	473
Главные решения	474
Описание элементов	475
Обработка XML-документа	479
Трехуровневая архитектура	479
Уровень данных	481
Уровень ввода	489
Уровень ввода: превращение XML в Species	491
Уровень вывода	497
Генерирование HTML-кода	510
Как самому построить приложение «Сорняки Эль Лимона»	535
Построение «Сорняков...» под Windows	535
Построение «Сорняков...» под UNIX	536
Заключение	537
Глава 13. Формат определения канала	538
Учебный пример на CDF-технологии	539
Исходная ситуация	539
Какую пользу принесет использование CDF-технологии	541
Создание CDF-файла	542
Тестирование CDF-файла	549
Присоединение содержания к CDF-файлу	550
Дальнейшее подсоединение страниц к CDF-файлу	555
Заключение	564
Приложение А. Языки и обозначения	565
Приложение В. XML-ресурсы и ссылки	569
Приложение С. Спецификация расширяемого языка разметки XML 1.0	574
Приложение D. XML-данные и типы данных DTD	622
Приложение Е. XML DTD для XML-данных	625
Приложение F. Свойства каскадных таблиц стилей CSS1	630

Приложение Г. Свойства каскадных таблиц стилей CSS2	639
Приложение Н. Поддержка читателей и список опечаток	652
Алфавитный указатель	659



Об авторах

Фрэнк Бумфрей (Frank Boumphrey) в настоящее время работает в фирме Cormorant Consulting, которая специализируется на создании баз данных в области медицины и юриспруденции. Программированием он начал заниматься еще в древние времена перфокарт и машинных языков. Одним из первых его заданий являлось написание программы, с помощью которой можно было отличить советскую межконтинентальную баллистическую ракету от стаи гусей. Судя по тому, что сейчас мы читаем об этом, программа получилась!

Устав мыслить шестнадцатеричными числами, он оставил программирование и получил диплом доктора медицины. Со временем Фрэнк Бумфрей стал профессором и возглавил отделение хирургии позвоночника в крупном институте на Среднем Западе. В дополнение к своим основным обязанностям Фрэнк занимался внедрением MRI в медицинскую практику. Выйдя на пенсию, он вернулся к своему первому увлечению – компьютерам и теперь пытается убедить медицинские учреждения хотя бы в какой-либо степени рационально организовывать поступающую информацию. С этой целью он читает медицинским работникам и администраторам системы здравоохранения лекции по основам электронного хранения и обработки документации.

Интересно, что наибольший успех Фрэнк Бумфрей имеет у юридических фирм, желающих рационализировать свои базы данных по медицине. Сейчас его главная цель – помочь XML стать основным языком Web-документов.

Оливия Диренцо (Olivia Direnzo) – художница, увлекается растениями и животными. Оливия окончила Корнелльский университет (Cornell) по специальности «зоология». Вместе с Полом Хоулом она работает в компании Honeylocust Media Systems, в которой занимается исследованием гиперсред, основанных на Web. Они оба считают, что это одно из интереснейших в настоящее время направлений.

Йон Дакетт (Jon Duckett), окончив Брюнельский университет (Brunel) в Лондоне и получив степень по психологии, возвратился домой, в Бирмингем, и поступил на работу в представительство издательства Wrox. Благодаря отличным ребятам из издательства Wrox он не слишком скучает по Лондону. Здесь, в Бирмингеме, Джон нашел точку приложения своих сил, стараясь в одиночку поддержать экономику нескольких стран, производящих чай.

Джо Грэф (Joe Graf) в течение шести лет работал консультантом по разработке крупномасштабных клиент-серверных систем, а затем основал компанию по производству программного обеспечения, получившую название E-comm Group

inc. Деятельность компании направлена на развитие языка XML как среды для совершения транзакций в Internet. В последнее время Джо возглавляет разработку и поддержку OFX Script, одного из первых серверов XML-приложений (XML Application Server), который позволит финансовым учреждениям осуществлять денежные транзакции с помощью Internet.

Дэйв Холэндер (Dave Hollander), после окончания Мичиганского технологического университета работал в лаборатории Bell, затем в нескольких компаниях, занимающихся компьютерной графикой, а также в Phillips DuPont Optical. В 1988 году Дэйв поступил в компанию Hewlett-Packard, где возглавил разработку издательских программ, производственных DTD и других систем. Дэйв создал сайт www.hp.com, и за тот период, что он управлял им, число посетителей сайта выросло с тысячи до миллиона в день. В настоящее время Дэйв занимается разработкой основанных на языке XML и предназначенных для связи с помощью каналов систем управления содержанием.

В течение всего этого времени Дэйв являлся образцовым мужем и отцом, возглавлял отряд молодых скаутов, председательствовал в школьном комитете, а также активно участвовал в формальной и неформальной деятельности в области определения стандартов, включая формат Rock Ridge (компакт-диски), Davenport и OSF DTD. Дэйв является действительным членом рабочей группы XML, соавтором спецификации по пространствам имен XML и сопредседателем рабочей группы консорциума W3C по XML-схемам.

Пол Хоул (Paul Houle) недавно получил в университете Cornell степень доктора натуральной философии по физике¹. В настоящее время работает в Германии, в Институте физики комплексных систем Макса Планка.

Пол отдает предпочтение операционной системе Linux и пишет программы на языках C, Perl и Java.

Тревор Дженкинс (Trevor Jenkins) – независимый консультант по системам управления текстами и документами. В числе его недавних клиентов можно было встретить и сеть ресторанов быстрого обслуживания, и главного издателя STM. До этого ему приходилось выполнять заказы для ведущих компаний в области телекоммуникаций и бухгалтерского дела. Язык SGML для него родной. Тревор участвует в работе BSI и ISO – ассоциаций, занимающихся внедрением SGML и относящихся к этому языку стандартов. В сферу их деятельности также входит техническая оценка языка XML. Тревор Дженкинс участвовал в подготовке нескольких технических отчетов по SGML, которые были опубликованы ведущей в области стандартизации организацией ISO («Методы использования SGML» и «Системы ввода текстовых данных»).

Питер Джоунс (Peter Jones) является техническим редактором и «домашним» автором издательства Wrox. Увлечений у него не счесть, однако с тех пор, как он начал работать в издательстве, времени ему хватает только на то, чтобы изредка погонять по окрестностям на огромном мотоцикле. Он хотел бы поблагодарить свою семью, друзей, всех сотрудников издательства Wrox и всех членов группы

¹ Это примерно соответствует российской степени кандидата физико-математических наук. (Прим.

XML-Dev, оказывавших ему поддержку при создании этой книги.

Эдриан Кингсли-Хьюз (Adrian Kingsley-Hughes, awkh@khd.co.uk) является техническим директором Kingsley-Hughes Development Ltd., консультационно-учебной фирмы, специализирующейся на языках для Web и языках визуального программирования. Эдриан основал эту фирму в 1996 году, когда изучал химию в Бангорском университете (Bangor). Кроме того, Эдриан – консультант в области развития Internet и программирования под Windows. Он создает обучающее программное обеспечение для таких далеких областей как химия, астрономия и валлийский язык. В свободное время пишет романы ужасов и изучает игру на диджеридо.

Кэти Кингсли-Хьюз (Kathy Kingsley-Hughes, kkh@khd.co.uk) работает управляющим директором в фирме Kingsley-Hughes Development Ltd. Со времени выхода в свет популярного сетевого журнала-мультфильма «Канал дракона» (The Dragon Channel), написанного на динамическом HTML, она увлеклась каналами CDF. Это издание журнала оказалось одним из первых активных каналов Internet в Великобритании. Перед этим Кэти в течение шести лет являлась инструктором по информационным технологиям, затем начала изучать химию и компьютеры в Бангорском и Открытом университетах. Она интересуется дистанционным обучением и Internet-классами. Кроме того, Кэти преподает химию, физику и математику.

Крэг Маккуин (Craig McQueen) – главный консультант компании Sage Information Consultants Inc. Он специализируется в разработке компонентов COM среднего уровня для электронной коммерции. Участвует в работе над рядом проектов в издательстве Wrox, а также каждый месяц пишет статьи для информационного бюллетеня Visual C++ Developer. В свое время Крэг возглавлял два проекта создания программного обеспечения для Internet: InContext FlashSite (Контекстный флэш-сайт) и InContext WebAnalyzer (Контекстный анализатор Web).

Крэг получил степень магистра в университете Торонто, где изучал взаимодействие человека и компьютера на факультете компьютерных наук. Итогом его исследований могут служить пять публикаций и три выступления на международных конференциях. Адрес электронной почты Крэга – cmcqueen@sageconsultants.com.

Стивен Мор (Stephen Mohr) начал заниматься программированием в школе – в те времена, когда это увлечение было не так популярно, как сейчас. Последние десять лет он специализировался в программировании для персонального компьютера. В качестве старшего архитектора систем программного обеспечения в фирме Omicron Consulting Стивен проектирует и создает системы, используя при этом языки C++, Java, JavaScript, COM, а также различные стандарты и протоколы Internet. Стивен имеет степени бакалавра и магистра компьютерных наук, полученные в политехническом институте Rensselaer. Занимается исследованием распределенных объектно-ориентированных систем и вопросами практического применения искусственного интеллекта.



Предисловие переводчика

В Internet появился новый язык. Зачем? Разве мало нам доброго старого HTML? Да, уже мало. То, что именно HTML способствовал бурному росту Internet, точнее, той его области, которая называется World Wide Web, давно стало общим местом в публикациях, посвященных Internet. Но теперь Сеть переросла своего «родителя» и требует таких возможностей, которых у него нет. Или почти нет.

HTML вполне справлялся с отображением некоторой информации в заранее заданной фиксированной форме. Но развитие Сети поставило новые задачи, для решения которых необходимо разделить данные и способ их представления. Вот простейший, но понятный всем пример. Много раз было сказано, что Internet по содержанию представляет собой сокровищницу, а по способу организации – свалку. Всякий, кто пользовался поисковыми машинами, знаком с проблемами информационного шума и поиска нужных сведений среди результатов поиска. Рутинная задача – найти нужную информацию – требует немалого искусства и затрат времени. Полностью автоматизировать ее не удастся. Почему? Именно потому, что HTML не разделяет данные и способ их отображения. Эту задачу решает новый язык – XML.

Язык XML появился совсем недавно и бурно развивается. Эта книга – одно из первых посвященных ему изданий на русском языке и уж почти наверняка первое из них, посвященное практической работе с XML. Как ни лестно идти в первых рядах, при этом возникают свои проблемы, и не последняя из них – перевод терминологии.

Сказать, что русская терминология в области XML не устоялась, значит ничего не сказать. В многочисленных публикациях, преимущественно переводных статьях обзорного характера, появляющихся в русскоязычной компьютерной прессе и русской части Internet, царит полнейший разноречие. Не только один и тот же термин переводится разными словами, это еще полбеды, но бывает и так, что одно и то же слово в разных статьях употребляется для перевода разных терминов. А некоторые переводы слова «entity», одного из ключевых терминов XML – «категория» или «сущность» – заставляют вспомнить лекции по диамату. В такой ситуации остается только уподобиться Шалтаю-Болтаю из «Алисы в Зазеркалье» и провозгласить: «Когда я беру слово, оно значит то, что я хочу». Все же мы старались избегать полного произвола и придерживались терминологии, применяемой может быть и не в самых многочисленных, но в наиболее интересных статьях. Вот краткая сводка переводов некоторых терминов XML, принятых в этой книге:

Well-formed document	(синтаксически) правильный документ
Valid document	состоятельный документ
Parser	анализатор
Validation	верификация
Validating parser	верифицирующий анализатор
Non-validating parser	неверифицирующий анализатор
Entity	компонент
Parameter entity	параметрический компонент
White spaces	пробельные литеры.

Возможно, манера изложения, принятая в этой книге, покажется отечественному читателю чересчур обстоятельной. Но учебный, по существу, характер книги, традиции издательства Wrox Publishers, в котором она выпущена, а также то, что книга посвящается относительно новому предмету, для которого трудно пока определить, какие сведения следует считать известными по умолчанию – все это вполне оправдывает некоторые длинноты и повторы.

Со времени выхода оригинального издания книги прошло достаточно времени, чтобы текущая ситуация, особенно в такой быстро развивающейся области, как язык XML, серьезно изменилась. Например, браузер Internet Explorer 5.0, на момент написания оригинала книги существовавший в виде бета-версии, в настоящее время уже доступен, в том числе и в локализованной русскоязычной версии. Можно только призвать читателя последовать совету, многократно повторяемому авторами книги: следите за новинками и загружайте обновленные версии приведенных кодов с сайта издательства Wrox, адрес которого неоднократно встречается на ее страницах.

Хотелось бы завершить это краткое вступление небольшим перечнем ссылок на ресурсы русского Internet, посвященные языку XML. Увы, пока это невозможно. Большая часть публикаций по XML в отечественной части Internet представляет собой переводы с английского статей обзорного характера или коллекции ссылок на англоязычные, по преимуществу, Internet-ресурсы (например, <http://citforum.indi.ru/internet/xml/links.shtml>). Особое место среди этих материалов занимает основательная и хорошо написанная статья Александра Печерского «Язык XML – Практическое введение». (Часть первая: <http://citforum.syzran.ru/internet/xml/index.shtml>. Часть вторая: <http://www.citforum.bonus.ru/internet/xml2/index.shtml>). Ее смело можно порекомендовать отечественному читателю для первого (и довольно основательного) знакомства с предметом.

Язык XML быстро развивается и со временем, вероятно, распространится так же широко, как и его предшественник HTML, хотя и навряд ли вытеснит его полностью (а почему – вы поймете, прочитав эту книгу). Помимо всех перечисленных в первой главе этой книги причин, по которым стоит изучать XML, есть еще одна – это очень интересно.

Виталий Глинка



Введение в XML

В наше время наиболее широко известной и впечатляющей новаторской разработкой в области Web-технологий можно считать XML, или расширяемый язык разметки (Extensible Markup Language). Надеемся, что, прочитав нашу книгу, читатели согласятся с этим утверждением. Те из них, кто уже освоил язык HTML, будут приятно удивлены гибкостью XML. Овладев этим способом разметки, вы не просто сэкономите время на создании приложений; вас несомненно поразит широта возможностей, которые XML предлагает вам как Web-разработчику.

Цель этой книги – подробно рассказать о языке XML и разнообразных методах, которые необходимо изучить для создания XML-приложений. Затем, на примере действующих XML-приложений, мы рассмотрим, как этот язык взаимодействует с некоторыми из новейших разработок Internet-технологий, такими как ActiveX Data Objects (объекты данных ActiveX, ADO) и Active Server Pages (активные серверные страницы).

Для кого написана эта книга

Настоящий сборник предполагает наличие у читателя базовых знаний по World Wide Web, а также знакомство с языком HTML, при этом читателю совсем необязательно иметь представление о языке XML. После изложения основ, необходимых для понимания XML и способов его использования, мы перейдем к конкретным задачам и постараемся научить вас создавать мощные XML-приложения. Последние главы отведены материалам, в которых используются скрипты Active Server Pages и SQL, языки Java и C++. Овладение подобными технологиями очень полезно для понимания некоторых примеров из последних глав, однако с архитектурой XML-приложений вы сможете разобраться, даже если не вполне владеете подобными новейшими приемами.

Эту книгу вам следует прочесть в том случае, если:

- ❑ вы хотите как можно больше узнать о языке XML;
- ❑ вам постоянно требуется больше тэгов, чем может предоставить HTML;
- ❑ вы хотите создавать свои собственные тэги;
- ❑ вы хотите, чтобы ваша разметка несла какой-то смысл;
- ❑ вы хотите разделить стиль и содержание документа;
- ❑ вы нуждаетесь в более мощном способе создания Web-приложений;
- ❑ вы не хотите остаться за бортом, когда в постоянно меняющемся мире Web начинается новая революция.

Что необходимо для работы с книгой

К созданию вашего первого XML-приложения вы можете приступить хоть сегодня. Для этого существует множество достаточно изощренных редакторов HTML- и XML-файлов, с помощью которых можно взяться за это дело, однако на первый раз вам вполне подойдет простейший текстовый редактор, например Windows Notepad (Блокнот). Если у вас есть компьютер, браузер типа Netscape Communicator 4.x или Internet Explorer 4.x, а также доступ к Internet для загрузки компонентов, необходимых при работе над примерами, сразу можете начинать писать на языке XML.

Прежде чем погрузиться в изучение языка XML, полезно взглянуть на саму разметку и историю ее развития в контексте языков SGML и HTML.

Что такое язык разметки

С разметкой, сами того не осознавая, мы сталкиваемся каждый день. Любой элемент документа, который придает тексту особый смысл или обеспечивает дополнительной информацией, можно отнести к разметке. Например, выделение текста является одним из способов привлечения внимания к тому или иному месту в документе.

Но подобное оформление текста не имеет никакого значения, если никто другой его не понимает. Таким образом, для пользования элементами разметки необходим набор правил, в которых надо:

- сформулировать, что именно является разметкой;
- точно описывать средства, с помощью которых она выполняется.

Всякий подобный набор называется языком разметки. Она бывает стилистическая, структурная и семантическая.

Стилистическая разметка

Этот вид показывает, как будет выглядеть документ. Примером стилистической разметки является использование в текстовом редакторе жирного шрифта или курсива. В языке HTML к стилистической разметке относятся тэги , <I>, .

Структурная разметка

Этот вид информирует о структуре документа. Тэги <Hn>, <P> и <DIV> являются примерами структурной разметки, которая указывает соответственно на заголовок, абзац и секцию-контейнер.

Семантическая разметка

Этот вид сигнализирует о содержании данных. Примерами семантической разметки являются тэги <TITLE> и <CODE> .

Языки разметки (markup languages) определяют правила, с помощью которых как раз и придается значение структуре и содержанию документов. Они описываются грамматикой и синтаксисом, в соответствии с которыми следует «говорить» на этом языке.

Тэги и элементы

Многие из тех, кто знаком с языком HTML, часто путают значения понятий *тэги* (tags) и *элементы* (elements). Тэги – это угловые скобки и текст между ними. Вот некоторые примеры использования тэгов в языке HTML:

`<P>` – тэг, отмечающий начало нового абзаца;

`<I>` – тэг, указывающий на то, что следующий за ним текст должен быть выведен курсивом;

`</I>` – тэг, указывающий конец части текста, которая должна быть выведена курсивом.

Под элементами же обычно понимают тэги в совокупности с их содержанием. Следующая конструкция является примером элемента:

`это текст, выделенный жирным шрифтом`

В обычных терминах тэги представляют собой значки, которые предписывают агенту пользователя (например, браузеру), как поступить с тем, что заключено в тэгах.

Уточнение *Агентом пользователя, или просто агентом (user agent), называется то, что действует от имени пользователя. Ваш начальник считает вас своим агентом; ваш компьютер является вашим агентом; ваш браузер является агентом вашего компьютера.*

Пустые элементы (не имеющие закрывающих тэгов), такие, например, как тэг `` в языке HTML, – в языке XML рассматриваются несколько иначе. Однако к этому вопросу мы вернемся позже.

На схеме, приведенной ниже (рис. 0.1), изображены части, из которых состоит элемент:

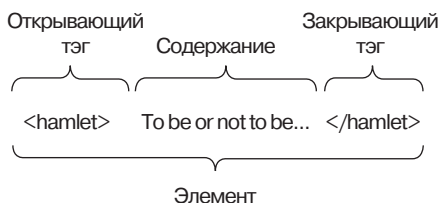


Рис. 0.1. Составные части элемента XML

Атрибуты

Любой тэг может иметь атрибут, если этот атрибут определен. Атрибуты принимают форму пар *имя/значение* (name/value) (их также называют парами *атрибут/значение* – attribute/value). Таким образом, каждому элементу может быть присвоен атрибут с именем. В то же время этот атрибут должен иметь некоторое значение. Тогда тэги принимают форму:

`<tagname attribute="value">` (`<имя_тэга атрибут="значение">`)

Например, в языке HTML 4.0 у элемента `<BODY>` могут быть следующие атрибуты:

CLASS	ID	DIR	LANG	STYLE	TITLE
BACKGROUND	BGCOLOR	ALINK	LINK	VLINK	TEXT

Таким образом, можно записать:

```
<BODY BGCOLOR="#000000" ALINK="#999999" LINK="#990099" VLINK="#888888" TEXT="#999999">
```

Языки разметки

В этом разделе мы рассмотрим три языка разметки: SGML, HTML и, конечно, XML. *SGML* – это метаязык, который используется для создания других языков разметки. Наиболее известным из них, написанным в стандарте SGML, является всеми нами любимый HTML, ведь именно он применяется в Web. Поскольку язык HTML разработан в соответствии с правилами SGML, его называют *приложением SGML*. Проблема практического употребления SGML состоит в том, что он очень сложен. Поэтому нам и интересен XML, созданный специально для Web как упрощенная версия SGML, сохраняющая большую часть его функциональных возможностей. Рассмотрим подробнее возможности каждого из упомянутых выше языков разметки.

Язык SGML

В 1986 году, задолго до того, как идея создания Web была воплощена в жизнь, *универсальный стандартизированный язык разметки SGML* (Standardized Generalized Markup Language) был утвержден в качестве международного стандарта (ISO 8879) определения языков разметки. Кстати, SGML существовал еще с конца шестидесятых. В ту пору его использовали для того, чтобы описывать языки разметки, при этом автору позволялось давать формальные определения каждому элементу и атрибуту языка. Таким образом, программисты имели возможность создавать свои собственные тэги, связанные с содержанием документа. В то время SGML был всего лишь одним из нескольких конкурирующих между собой подобных языков, однако популярность одного из его потомков – HTML – дала SGML неоспоримое преимущество перед своими собратьями.

Как язык SGML является очень мощным средством. Однако вместе с мощностью пришла и сложность, поэтому все его широкие возможности используются редко. Кроме того, SGML-документ трудно интерпретировать без определения языка разметки, которое хранится в *определении типа документа* (Document Type Definition, DTD). В DTD сгруппированы все правила языка в стандарте SGML. DTD необходимо посылать вместе с SGML-документом или включать в документ для того, чтобы можно было распознать тэги, созданные пользователем. Языки разметки, созданные в стандарте SGML, известны как *SGML-приложения*.

Язык HTML

Первоначально язык HTML был всего лишь одним из SGML-приложений. Он описывал правила, по которым должна быть подготовлена информация для World Wide Web. Таким образом, язык HTML – это набор предписаний SGML, сфор-

мулированных в виде определения типа документа (DTD), которые объясняют, что именно обозначают тэги и элементы. В случае языка HTML, DTD хранится в браузере, оно описано во множестве книг, а также на нескольких Web-сайтах. По размеру язык HTML во много раз меньше языка SGML. В то же время он намного проще и легче для изучения, что добавило ему популярности. Сейчас HTML принят во всех кругах компьютерного сообщества.

Язык HTML как способ разметки технических документов был создан Тимом Бернерсом-Ли (Tim Berners-Lee) в 1991 году специально для научного сообщества. С его помощью оказалось возможным значительно упростить организацию специальных текстов и передачу их через компьютеры различного типа. Идея состояла в создании набора особых словесных формул, которые можно было употреблять для разметки документов. Применение подобных формул должно было обеспечить передачу документов между компьютерами таким образом, чтобы адресаты могли воспроизводить документ в удобном формате. Например:

```
<H1> Это заголовок первого уровня </H1>  
<H2> Это заголовок второго уровня </H2>  
<PRE> Это текст, для которого важно сохранить форматирование </PRE>  
<P> Текст между этими тэгами образует абзац </P>
```

В те далекие времена представители научного сообщества почти не обращали внимания на внешний вид посылаемых и получаемых документов. Ученым было важно сохранить смысл передаваемого текста. Их не волновали такие мелочи, как цвет шрифта или точный размер заголовка первого уровня.

Для передачи информации по Internet язык HTML использует так называемый *протокол передачи гипертекстов* (Hypertext Transfer Protocol, HTTP). Это только один из протоколов, используемых в Internet, входящий в широко известный *набор протоколов Internet* (Internet Protocol Suite), который чаще называют *TCP/IP*. В настоящее время широко используются и несколько других протоколов из набора TCP/IP. До того как появился язык HTML, самым популярным был *протокол передачи файлов* (File Transfer Protocol, FTP).

Преимущество перед другими протоколами дала HTTP легкость, с которой он мог быть использован для подключения к другому документу. Объединение этого протокола с простым для изучения языком обеспечило быстрое распространение систем, реализующих язык HTML и протокол HTTP.

Однако по мере того, как HTML приобретал все более широкое распространение и Web-браузеры становились все доступней, пользователи, не входившие в научное сообщество, стали в массовом порядке создавать свои собственные страницы. Эти представители «ненаучных» кругов все чаще начали обращать внимание на внешний вид своих материалов. Производители браузеров, используемых для просмотра Web-сайтов, с готовностью предлагали различные тэги, которые позволяли авторам Web-страниц представлять свои документы в куда более разнообразном виде, чем просто текст ASCII. Первой на этот путь встала компания Netscape, которая добавила знакомый нам тэг ``, позволявший пользователям менять как сами шрифты, так и их размер и ширину. С этого начался быстрый рост числа тэгов, поддерживаемых браузерами.

С новыми тэгами пришли новые проблемы. Различные браузеры воспроизводили новые тэги по-разному. Сегодня существуют Web-сайты, на которых специально указывается: «Лучше всего просматривать в Netscape Navigator» или «Сделано для Internet Explorer», и при всем этом от пользователей ожидают, что созданные ими Web-страницы будут похожи на документы, оформленные в самых совершенных настольных издательских системах.

Таким образом, потенциал браузера как новой платформы для приложений был признан очень быстро, и Web-разработчики приступили к созданию распределенных прикладных систем для бизнеса, используя Internet в качестве среды для получения информации и осуществления финансовых транзакций.

Недостатки языка HTML

В связи с широким распространением языка HTML все больший круг пользователей вовлекался в процесс написания HTML-приложений. Их усилия в первую очередь были направлены на увеличение числа и сложности операций, осуществляемых в Web. В результате скоро стали очевидны слабые места языка HTML, а именно:

- HTML имеет *фиксированный набор тэгов*. Вы не можете создавать свои тэги, понятные другим пользователям;
- HTML – это исключительно *технология представления данных*. HTML не несет информации о значении содержания, заключенного в тэгах;
- HTML – «*плоский*» язык. Значимость тэгов в нем не определена, поэтому нельзя представить иерархию данных;
- *в качестве платформы для приложений используются браузеры*. HTML не обладает достаточной мощностью для создания Web-приложений на том уровне, к которому в настоящее время стремятся Web-разработчики. Например, на языке HTML достаточно сложно написать приложение для профессиональной обработки и поиска документов;
- *большие объемы трафика сети*. Существующие HTML-документы, используемые как приложения, засоряют Internet большими объемами трафика в системах клиент-сервер. Примером может служить пересылка по сети большой группы записей общего характера, в то время как необходима только небольшая часть этой информации.

Со временем усилиями пользователей, пытавшихся представить информацию в самых разнообразных формах, Web становилась все более и более фрагментированной. Создавая свои страницы, авторы пытались использовать не только разные реализации языка HTML, но и привлекали языки скриптов, *динамический HTML* (Dynamic HTML), *каналы* (Channels) и другие технологии, которые оказались несовместимы с многими браузерами.

Вот так и случилось, что, с одной стороны, язык HTML являлся очень удобным средством разметки документов для использования в Web, а с другой – документ, размеченный в HTML, нес мало информации о своем содержании, и это в то время, когда использование документа в деловых целях требовало серьезных знаний о его сути. Если тот или иной документ несет достаточно полную информацию о своем содержании, появляется возможность сравнительно легко провести автоматичес-

Конец ознакомительного фрагмента.

Приобрести книгу можно
в интернет-магазине «Электронный универс»
(e-Univers.ru)