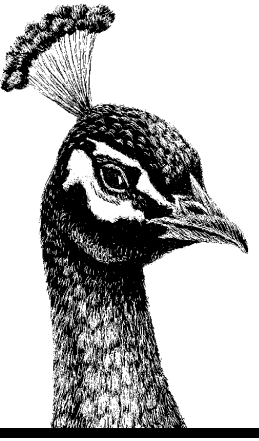
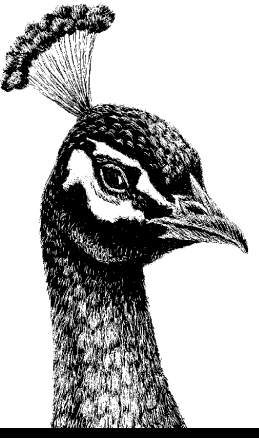


XML IN A NUTSHELL

Elliotte Rusty Harold & W. Scott Means

O'REILLY®



XML

СПРАВОЧНИК

Эллиот Расти Гарольд
и У. Скотт Минс

Санкт-Петербург
2002

Эллиот Расти Гарольд и У. Скотт Минс

XML. Справочник

Перевод Л. Фрейдина

Главный редактор	*А. Галунов*
Зав. редакцией	*Н. Макарова*
Научный редактор	*Е. Морозов*
Редактор	*Р. Павлов*
Корректура	*С. Беляева*
Верстка	*М. Мышкина*

Гарольд Э., Минс С.
XML. Справочник. – Пер. с англ. – СПб: Символ-Плюс, 2002. – 576 с., ил.
ISBN 5-93286-025-1

«XML. Справочник» необходим каждому серьезному разработчику, использующему эту новую технологию. Читатели найдут различные темы – от базовых синтаксических правил до деталей создания DTD или API для чтения и записи XML-документов на разных языках программирования. Поняв суть базового стандарта XML, вы сможете быстро разобраться в тонкостях DTD, пространств имен, соблюдения корректности XML-документов и поддержки Unicode. Обзор ключевых технологий поможет получить практические знания по XSLT, XPath, XLink, XPointer, CSS и XSL-FO.

Наверняка многие заинтересуются применением XML для обработки данных. Сейчас XML все шире применяется для работы со структурированными документами: электронными и математическими таблицами, статистическими и финансовыми отчетами и форматами файлов программного обеспечения. Рассматриваются утилиты и API, такие как SAX и DOM, необходимые для написания программ обработки XML. Книга содержит справочные главы, в которых приведены подробные синтаксические правила и примеры использования DTD, XPath, XSLT, SAX и DOM. Здесь можно быстро найти определенный синтаксис, с которым вы знакомы, однако не помните в точности.

Углубившись в изучение XML, вы непременно пожелаете иметь эту книгу под рукой. Она окажет вам неоценимую помощь в правильном форматировании файлов и структур данных для XML-документов.

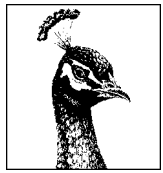
ISBN 5-93286-025-1
ISBN 0-596-00058-8 (англ)

Издательство «Символ-Плюс». 193148, Санкт-Петербург, ул. Пинегина, 4,
тел. (812) 324-5353, edit@symbol.ru. Лицензия ЛП N 000054 от 25.12.98.

Налоговая льгота – общероссийский классификатор продукции
ОК 005-93, том 2; 953000 – книги и брошюры.

Подписано в печать 30.01.2002. Формат 70x100$^{1}/_{16}$.
Печать офсетная. Объем 36 печ. л. Тираж 2000 экз. Заказ N
Отпечатано с диапозитивов в ФГУП «Печатный двор» им. А. М. Горького Министерства РФ по делам печати, телерадиовещания и средств массовых коммуникаций.
197110, Санкт-Петербург, Чкаловский пр., 15.

Оглавление



Предисловие

XML – это одна из наиболее важных в истории информатики разработок в области синтаксиса документов. За последние несколько лет XML был принят в самых разных сферах деятельности: в законодательстве, аэронавтике, финансах, страховании, робототехнике, мультимедиа, благотворительности, туризме, искусстве, строительстве, телекоммуникациях, разработке программного обеспечения, сельском хозяйстве, физике, журналистике, теологии, торговле и исследованиях по средневековой литературе. XML стал синтаксисом новых форматов документов практически во всех областях применения компьютеров. Он используется в Linux, Windows, Macintosh и на других платформах. Мейнфреймы на Уолл Стрит продают и покупают акции, обмениваясь XML-документами. Дети, играя дома в компьютерные игры, сохраняют документы в XML. В формате XML информация о счете в матчах приходит спортивным болельщикам в реальном времени на сотовые телефоны. XML – это просто самый ясный, надежный и гибкий синтаксис документов из всех изобретенных.

Эта книга – самый надежный путеводитель по стремительно развивающемуся миру XML. В ней освещаются все аспекты XML, начиная с простейших синтаксических правил и заканчивая подробностями создания DTD и API для чтения и записи XML-документов с помощью различных языков программирования.

О чем рассказывается в этой книге

Существуют сотни официально утвержденных XML-приложений от W3C и других разработчиков стандартов, таких как OASIS и Object Management Group. Еще больше неофициальных, не стандартизиро-

ванных приложений, созданных частными лицами или корпорациями, например Channel Definition Format от Microsoft и Mind Reading Markup Language от Джона Гвархардо (John Guarjardo). В этой книге нельзя рассказать обо всех, так же как в книге по Java невозможно рассмотреть все программы, которые уже написаны или могут быть написаны на Java. В нашей книге главное внимание уделяется собственно XML. В ней рассматриваются основополагающие правила, которым должны соответствовать все XML-документы и следовать их создатели, будь то веб-дизайнеры, создающие на своих страницах анимацию с помощью SMIL, или программисты C++, применяющие SOAP для сохранения объектов в удаленной базе данных.

В книге также описаны общие вспомогательные технологии, представляющие собой надстройки XML и используемые в самых различных XML-приложениях. Эти технологии включают:

XLinks

Синтаксис для гиперссылок между XML- и неXML-документами, основанный на атрибутах. Он определяет простые однонаправленные ссылки, знакомые всем по HTML, многонаправленные ссылки между несколькими документами, а также связи между документами, к которым отсутствует доступ на запись.

XSLT

XML-приложение, которое описывает преобразования одного документа в другой в рамках того же или другого XML-словаря.

XPointers

Синтаксис для обозначения определенных частей XML-документа в ссылках URI; часто используется вместе с XLink.

XPath

НеXML-синтаксис, используемый XPointers и XSLT для обозначения определенных частей XML-документа. Например, с помощью XPath можно определить местоположение в документе третьего элемента типа `address` или всех элементов с атрибутом `email`, равным `elharo@metalab.unc.edu`.

Namespaces (пространства имен)

Средство для установления различия между элементами и атрибутами из разных XML-словарей, имеющими одинаковые имена; например, между заголовком книги и заголовком веб-страницы о книгах.

SAX

Простой API для XML, событийно-ориентированный интерфейс прикладного программирования для Java, реализуемый многими XML-анализаторами.

DOM

Объектная модель документа (Document Object Model) – API, ориентированный на представление документа в виде дерева, рассматривает документ XML как набор вложенных объектов с различными свойствами.

Все эти технологии, определяемые с помощью XML (XLinks, XSLT и пространства имен) или другого синтаксиса (XPointer, XPath, SAX и DOM), используются в самых разных приложениях XML.

В книге специально не рассматриваются приложения XML, которые заинтересуют лишь некоторых пользователей. К ним относятся:

SVG

Scalable Vector Graphic (масштабируемая векторная графика) – одобренный W3C стандарт, который применяется для кодирования векторных рисунков в XML.

MathML

Mathematical Markup Language (язык математической разметки) – одобренное W3C стандартное приложение XML, используемое для вставки математических выражений в веб-страницы и другие документы.

CML

Chemical Markup Language (язык химической разметки) – это одно из первых приложений XML, используемое при описании формул в химии, физике твердого тела, молекулярной биологии и других науках.

RDF

Resource Description Framework (система описания ресурсов) – стандартизированное W3C XML-приложение, которое применяется для описания ресурсов, в особенности для метаданных, таких как библиотечные карточки.

CDF

Channel Definition Framework (система описания каналов) – это нестандартное XML-приложение, разработанное Microsoft, с помощью которого можно публиковать веб-сайты в Internet Explorer для оффлайнового просмотра.

Иногда мы приводим примеры из одного или нескольких таких приложений, но не рассматриваем глубоко все аспекты соответствующего словаря. Будучи интересными и важными, эти приложения (как и сотни других подобных) требуют в первую очередь специального программного обеспечения, предназначенного для конкретного формата. Например, дизайнеры графики не работают напрямую с SVG. Вместо этого они создают документы в формате SVG с помощью своих при-

вычных средств, таких как Adobe Illustrator. Они могут даже не знать о том, что пользуются XML.

В этой книге в основном рассматриваются стандарты, значимые для большинства разработчиков, работающих с XML. Мы будем изучать технологии XML, которые охватывают широкий диапазон приложений, а не те, что важны лишь в некоторых сферах деятельности.

Структура книги

Часть I «Понятия XML» представляет собой введение в основные стандарты, образующие прочный фундамент, на котором строятся все XML-приложения и программное обеспечение. Вы максимально быстро изучите такие понятия, как корректный XML, DTD, пространства имен и Unicode.

В части II «Повествовательные документы» раскрываются технологии, применяемые главным образом в повествовательных XML-документах, таких как веб-страницы, книги, статьи, дневники и сценарии. Вы узнаете об XSLT, CSS, XSL-FO, XLinks, XPointers и XPath.

Большой неожиданностью для XML стал тот энтузиазм, с которым его стали использовать для структурированных документов с большим количеством данных, таких как электронные таблицы, финансовая статистика, математические таблицы и форматы файлов для программного обеспечения. Часть III «XML для данных» посвящена применению XML в подобных документах. Здесь детально рассматриваются утилиты и API, которые необходимы, чтобы писать программы для обработки XML, в том числе SAX (Simple API for XML, простой API для XML), и DOM (Document Object Model, объектная модель документа) консорциума W3C.

И наконец, часть IV «Справочник» – серия справочных глав, основа любой книги нашей серии. В этих главах подробно даются синтаксические правила для основных технологий XML, в том числе XML, DTD, XPath, XSLT, SAX и DOM. Если вы случайно что-то забыли, обращайтесь к этой части, чтобы быстро найти нужный синтаксис.

Условные обозначения, принятые в книге

`Моноширинный шрифт` используется для обозначения:

- Примеров и фрагментов кода.
- Всего, что может встретиться в XML-документе, в том числе имен элементов, тегов, значений атрибутов, ссылок на сущности и инструкций обработки.
- Всего, что может присутствовать в программе, в том числе ключевых слов, операторов, имен методов, имен классов и литералов.

`Моноширинный жирный шрифт` используется для:

- Обозначения пользовательского ввода.
- Выделения фрагмента текста.

`Моноширинный курсив` используется для обозначения:

- Заменяемых элементов в строках кода.

Курсив используется для обозначения:

- Новых терминов – там, где они впервые определяются.
- Имен каталогов, файлов и программ. (Однако если имя программы также является именем Java-класса, оно печатается моноширинным шрифтом, как и другие имена классов.)
- Имен хостов и доменов (*www.xml.com*).
- URL (*http://ibiblio.org/xml/*).

Значительные фрагменты кода, полные тексты программ и документы обычно печатаются в виде отдельного абзаца, например:

```
<?xml version="1.0"?>
<?xml-stylesheet href="person.css" type="text/css"?>
<person>
  Алан Тьюринг
</person>
```

XML чувствителен к регистру. Элемент PERSON – это не то же самое, что элемент person или Person. Языки, чувствительные к регистру, не всегда позволяют авторам придерживаться правил английской грамматики. Если есть возможность переписать предложение так, чтобы избежать конфликта, мы всегда будем стараться это делать. Однако в некоторых случаях, когда никак нельзя обойти эту проблему, наш выбор будет не в пользу традиционного английского языка.

И наконец, хотя большинство примеров, приводимых в этой книге, можно назвать детскими, и вряд ли стоит их использовать повторно, но некоторые все же представляют реальную ценность. Вы можете свободно включать их или любые их части в собственный код. Никакого специального разрешения не требуется. Мы считаем, что эти примеры являются всеобщим достоянием (хотя это ни в коей мере не относится к пояснительному тексту).

Просьба комментировать

Мы рады услышать от читателей общие замечания и предложения о том, как можно улучшить эту книгу, какие темы осветить, а также конкретные исправления. Вы можете обращаться к авторам по адресам электронной почты *elharo@metalab.unc.edu* и *smeans@enterprise-webmachines.com*. Однако просим иметь в виду, что каждый из нас по-

лучает несколько сотен писем в день и не может отвечать на каждое лично. Поэтому, чтобы повысить шансы на получение персонального ответа, пожалуйста, укажите, что вы являетесь читателем данной книги. Мы также просим отправлять сообщения с той учетной записи, на которую вы хотите получить ответ. Убедитесь, кроме того, что ваш обратный адрес установлен правильно. Нет ничего более обидного, чем, потратив час или больше на поиск и написание подробного ответа на интересный вопрос, получить лишь уведомление о невозможности отправки из-за того, что корреспондент отправил письмо с публичного терминала и не удосужился установить параметры броузера, чтобы включить действительный адрес электронной почты.

Информация в этой книге тестировалась и проверялась, но, возможно, в чем-то могли произойти изменения (или даже обнаружатся ошибки). Мы придерживаемся проверенного принципа: «Если вам нравится книга, расскажите друзьям. Если не нравится, расскажите нам». Мы особенно заинтересованы в информации об ошибках. Несмотря на кропотливую работу авторов и редакторов над этой книгой, наверняка осталось некоторое количество ошибок и опечаток, которые мы пропустили. Если вы найдете ошибку или опечатку, сообщите нам об этом, чтобы мы могли ее исправить. Информацию о любых ошибках, а также пожелания для будущих изданий можно присылать по адресу:

O'Reilly & Associates, Inc.
101 Morris Street
Sebastopol, CA 95472
1-800-998-9938 (in the United States or Canada)
1-707-829-0515 (international/local)
1-707-829-0104 (fax)

У этой книги есть веб-сайт, где приводятся поправки, примеры и другая дополнительная информация. Этот сайт находится по адресу:

http://www.oreilly.com/catalog/xmlnut

Перед тем как сообщать нам об ошибках, проверьте, нет ли там уже соответствующего исправления. Технические вопросы и отзывы о книге присылайте по адресу:

bookquestions@oreilly.com

Дополнительную информацию о наших книгах, конференциях, программном обеспечении, собраниях ресурсов (Resource Centers) и о сети O'Reilly (O'Reilly Network) смотрите на нашем сайте:

http://www.oreilly.com

Благодарности

Многие люди участвовали в создании этой книги. Заварил всю эту кашу первый редактор, Джон Познер (John Posner); он сделал много по-

лезных замечаний, которые значительно улучшили книгу. Когда Джон переключился на другую работу, дело было завершено Лори Петрицки (Laurie Petrycki). Стивен Спейнауэр (Stephen Spainhour) заслуживает особой благодарности за работу над справочным разделом. Его усилия по организации и просмотру материала помогли улучшить книгу. Хотим поблагодарить Мэтта Сарджента (Matt Sergeant) и Дидье Мартина (Didier P. H. Martin) за их тщательный технический просмотр материала и толковые предложения.

Мы также хотим поблагодарить тех, без кого нам было бы не о чем писать: всех, чьими стараниями XML добился за последние несколько лет такого успеха. Этих людей так много, что мы можем здесь перечислить лишь некоторых. Мы благодарим (в алфавитном порядке) Тима Бернерса-Ли (Tim Berners-Lee), Йона Босака (Jon Bosak), Тима Брея (Tim Bray), Джеймса Кларка (James Clark), Чарльза Гольдфарба (Charles Goldfarb), Джейсона Хантера (Jason Hunter), Майкла Кэя (Michael Kay), Бретта Маклахлина (Brett McLaughlin), Дэвида Меггинсона (David Megginson), Дэвида Очарда (David Orchard), Уолтера Пери (Walter E. Perry), Саймона Сент-Лорента (Simon St. Laurent), С. М. Сперберга-МакКуина (C. M. Sperberg-McQueen), Джеймса Таубера (James Tauber), Б. Томми Асдина (B. Tommie Usdin) и Марка Вутку (Mark Wutka). Приносим извинения всем, кого мы неумышленно пропустили.

Эллиотт хотел бы поблагодарить своего агента, Дэвида Рогелберга (David Rogelberg), который убедил его в том, что вместо работы в офисе можно зарабатывать на жизнь написанием книг, подобных этой. Вся команда Sunsite (теперь это *ibiblio.org*) также помогала ему самыми разными способами налаживать контакты с читателями в течение последних нескольких лет. Все эти люди заслуживают огромной благодарности и уважения. И наконец, как всегда, больше всех он благодарен своей жене Бет, без любви и поддержки которой эта книга никогда бы не появилась.

Скотт больше всего хотел бы поблагодарить свою прекрасную жену Селию, которой слишком долго пришлось оставаться «компьютерной вдовой». Он также благодарит свою дочь Селен, которая понимает, почему папочка не может поиграть с ней, когда работает, и Скайлера – просто за то, что он есть. Кроме того, он хочет поблагодарить команду Enterprise Web Machines, без которой не нашлось бы времени для написания книги. И наконец, он благодарит Джона Познера, который вовлек его в это дело, и Лори Петрицки – за последующую совместную работу.

Эллиот Расти Гарольд
elharo@metalab.unc.edu

У. Скотт Минс
smeans@enterprisewebmachines.com

Глава 14

XML как формат данных

Несмотря на то что истоки XML находятся в мире документов, он развивается как формат данных, наилучшим образом подходящий для передачи и хранения информации в Интернете. Как стандарт ASCII сделал написанный латинскими буквами текст переносимым между несопоставимыми системами, так XML обещает сделать сложные структуры данных переносимыми между различными языками программирования и компьютерными архитектурами. Хотя базовая структура языков остается той же, использование XML для описания программных данных, а не предназначенных для чтения документов, предлагает уникальный комплекс задач. В этой главе перечислены наиболее распространенные XML-приложения, ориентированные на данные, и приведен обзор утилит и технологий, используемых для их реализации.

Приложения XML для программистов

До XML отдельные программисты должны были сами определять, как форматировать данные при необходимости их сохранить или передать. В большинстве случаев данные не предназначались для использования вне исходной программы, и потому программисты хранили их в наиболее удобном формате, который они только могли придумать. За эти годы развилось несколько стандартов de facto (RTF, CSV и вездесущий формат *ini*-файлов Windows), однако данные, записанные одной программой, обычно могли быть прочитаны только этой же программой. Фактически данные могла прочитать зачастую только конкретная *версия* той же программы.

Быстрое распространение XML и бесплатных XML-утилит среди программистского сообщества дало разработчикам очевидное решение в выборе для своих приложений формата хранения или передачи данных. Для всех приложений, кроме самых тривиальных, преимущества использования XML для хранения и извлечения данных значительно перевешивают дополнительные накладные расходы на включение XML-анализатора в приложение. Уникальные преимущества XML как формата данных для программного обеспечения включают:

- Простой синтаксис: легкость генерации и обработки.
- Поддержка вложенности: программы при помощи тегов с легкостью могут представлять структуры с вложенными элементами.
- Легкость отладки: читабельный формат данных легко воспроизвести простым текстовым редактором.
- Независимость от языка и платформы: XML и Unicode гарантируют, что файл данных будет переносим практически на любую существующую сегодня популярную компьютерную архитектуру и комбинацию языков.

Кроме этих основных преимуществ XML иногда действительно позволяет создавать новые типы приложений, которые ранее были невозможны (или очень дороги) в реализации.

Смешанные среды

Современные корпоративные приложения часто требуют работы программного обеспечения на самых разных компьютерных системах под разными ОС. Выбор коммуникационного протокола включает в себя поиск общего знаменателя, доступного на всех системах. При большом количестве XML-анализаторов, которые могут быть свободно интегрированы в приложение, XML становится популярным форматом для обеспечения общего доступа к корпоративным данным.

Представьте себе типичное корпоративное приложение, которое должно отображать данные из мейнфрейма пользователям, соединяющимся с веб-сайтом предприятия. В этом случае XML действует в качестве «клея» для соединения веб-сервера с устаревшим приложением на мейнфрейме. Простое приложение с XML-интерфейсом получает запросы от веб-сервера, вызывает старое приложение и преобразует результат его работы в XML. Используя такую технологию, как XSLT, веб-сервер может затем преобразовать XML в любой из приемлемых сетевых форматов. Приняв XML в качестве единого языка для вашего предприятия, вы значительно облегчаете использование существующих данных в новых технологиях.

Коммуникации

Построение гибких коммуникационных протоколов, соединяющих несопоставимые системы, всегда было одной из главных задач информатики. С распространением компьютерных сетей и Интернета важность построения распределенных систем стала еще более значительной. Во многом так же, как протокол HTTP позволяет веб-клиентам просматривать содержимое веб-серверов, такие стандарты, как Simple Object Access Protocol (простой протокол доступа к объектам, SOAP), нацелены на то, чтобы программные службы были доступны любой клиентской программе в Интернете.

Сериализация объектов

Как и в отношении коммуникаций, на вопрос о том, где и как хранить состояние перманентных объектов, в разное время давались разные ответы. С популяризацией объектно-ориентированных языков, таких как C++ и Java, язык и среда времени выполнения часто имеют дело с техникой сериализации объектов. К сожалению, многие из этих технологий возникли до появления XML.

Большинство существующих методов сериализации в значительной степени зависят от языка и архитектуры. Сериализованный объект часто хранится в двоичном формате, который человек не может прочесть. Эти файлы не решают проблем с искажениями и совместимостью, так как структура объекта часто меняется и требует отдельной работы со стороны программиста.

Возможности, благодаря которым XML стал популярен в качестве коммуникационного протокола, делают его не менее популярным и в качестве формата сериализации содержимого объектов. Облегчается просмотр содержимого объекта, редактирование вручную и даже восстановление испорченных файлов. Гибкая природа XML позволяет формату файлов бесконечно развиваться, при этом поддерживая обратную совместимость с более старыми версиями.

Хранение и извлечение данных

В приложении граница между XML-файлом и реляционной базой данных часто размыта. Хотя XML слишком многословен и замедляет поиск в высокопроизводительных приложениях баз данных, его можно использовать в качестве простого, самодостаточного хранилища для небольших баз данных.

В сочетании с DHTML файлы данных XML используются для разгрузки веб-серверов, перенося приложения, связанные с нетранзакционным поиском и извлечением данных, на клиентский веб-броузер. На стороне сервера XML используется в качестве альтернативного механизма доставки результатов запросов. Такие продукты, как Microsoft

SQL Server 2000 и Oracle8i, включают собственную поддержку XML как формата извлечения данных.

Описание данных

С самого начала основное назначение XML состояло в том, чтобы дать пользователям возможность недвусмысленно определять новые форматы данных и совместно использовать эти форматы с другими. Механизм DTD обеспечивает однозначный способ отождествления документов с данными, полученными из той же предметной области. Гарантируя корректность и действительность XML-документа (в отношении конкретного DTD), любое приложение, получающее данные, соответствующие одному и тому же DTD, может немедленно их воспринять. Рассмотрим следующий пример:

```
<?xml version="1.0"?>
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 1.0//EN"
               "http://www.w3.org/TR/REC-smil/SMIL10.dtd">
<smil>
       <head>
       <layout>
           <root-layout width="300" height="200" background-color="white" />
           <region id="text_region" left="75" top="50" width="150" he-
ight="100" />
       </layout>
       </head>
       <body>
           <text region="text_region" src="Hello.txt" dur="3s"/>
           <text region="text_region" src="World!!!.txt" begin="1s"
dur="3s"/>
       </body>
</smil>
```

Объявление `DOCTYPE` указывает на то, что этот документ соответствует версии 1.0 спецификации Synchronized Multimedia Integration Language (SMIL) и что любой проигрыватель, поддерживающий SMIL, сможет его интерпретировать. Абсолютный URL *http://www.w3.org/TR/REC-smil/SMIL10.dtd* указывает на копию официального DTD SMIL, находящуюся на сайте W3C. SMIL DTD гарантирует, что этот документ соответствует официальному стандарту SMIL.

«Действительный» не значит правильный

Важно помнить, что ограничения, которые DTD накладывает на XML-документ, не являются особенно строгими. Большинство приложений имеют собственные ограничения предметной области, которые следует соблюдать. Кроме всего прочего, DTD *может* воздействовать на:

- Отношения между элементами
- Типы, имена и возможные значения атрибутов элемента
- Содержимое элемента

Однако оно не может принудить использовать правильные типы данных элемента, размеры и значения. Например, следующий отрывок из XML является действительным относительно своего DTD:

```
<integer_val>Неверное значение</integer_val>
```

Однако для приложения, ожидающего целую величину, этот документ определенно не допустим. Присущая XML 1.0 неопределенность является предметом работы рабочей группы по XML Schema.

Resource Description Framework (RDF)

На текущей стадии эволюции Сети большая часть интернет-трафика – это данные, передаваемые между потребителями, использующими веб-броузеры, и контент-провайдерами (веб-серверами). По мере того как предприниматели будут проводить все большую часть своих ежедневных операций в онлайне, это разделение между производителем и потребителем начнет смещаться. Resource Description Framework (структура описания ресурсов, RDF) предназначена для поддержания роста равноправных сервисов «компьютер-компьютер».

RDF по существу определяет язык, используемый для описания предоставляемых объектов и сервисов, в предназначенном для машинного чтения формате. XML-приложение включает средства описания объектов, их свойств и отношений друг с другом. Предполагаемый итог развития RDF состоит в том, что программные приложения будут иметь возможность единообразно и недвусмысленно описывать свои информационные продукты, службы и содержимое для использования «червяками» и автономными агентами.

Схемы XML

Язык XML Schema предназначен для того, чтобы дополнить основной механизм DTD, включенный в XML 1.0, значительно более строгой

системой объявления структуры и содержимого XML-документов. Кроме основных средств описания связей между элементами и атрибутами, встроенных в XML 1.0, схемы позволяют создателям приложений налагать определенные ограничения по типам данных на содержимое своих документов. Они также предоставляют поддержку создания сложных пользовательских типов данных, диапазонов и масок. Вместе с такими стандартами, как RDF, XML Schema предназначен для того, чтобы помочь разработчикам в создании машинно-ориентированных служб, к которым можно осуществлять доступ из Интернета.

Рассмотрим следующий фрагмент XML:

```
<!ELEMENT birthday (#PCDATA)>
. . .
<birthday>Green</birthday>
```

Так как XML 1.0 не поддерживает включение семантической информации о формате символьных данных, XML-анализатор не знает, что значение `Green` должно было быть полем даты. Новый язык XML Schema позволяет включать в информацию о структуре дополнительные сведения о типе символьных данных:

```
<element name="birthday" type="date">
```

Если ошибочный элемент из предыдущего примера включен в документ, в котором присутствует это правило, XML-анализатор может предупредить о нарушении типа данных, заданного в определении элемента `birthday`.

Средства для программистов

XML сам по себе не является ничем, кроме пассивного формата, применяемого для кодирования данных. Чтобы воспользоваться им, программисты должны включить в свои приложения технологию генерации и анализа XML. К счастью, относительно простая грамматическая структура XML сделала доступными множество отличных и обычно бесплатных библиотек. Несмотря на широкое разнообразие языков и платформ, поддерживаемых этими анализаторами, большинство предоставляют доступ к XML-данным с помощью одной из двух моделей API: Document Object Model (объектная модель документа, DOM) или Simple API for XML (простой API для XML, SAX). Глубокое понимание этих двух технологий поможет разработчикам выбрать верные средства для своего приложения. Глава 16 «Объектная модель документа (DOM)» и глава 17 «SAX» более подробно исследуют эти две технологии.

Объектная модель документа (DOM)

Как и HTML, XML – это иерархический формат данных. Элементы данных могут быть вложены в другие элементы данных, и к ним могут присоединяться атрибуты. Объектная модель документа определяет иерархию объектов, которая может полностью описать документ XML или HTML как рекурсивный список списков. Наиболее важной чертой DOM является то, что для создания этой структуры данных весь документ должен быть проанализирован и сохранен в памяти. Это создает затруднения в случае очень больших документов или приложений, когда документ может быть не доступен целиком к тому моменту, когда должна производиться обработка. Кроме того, многие реализации DOM требовательны к объему памяти, а некоторые приводят к 100-процентному использованию памяти сверх исходного размера документа. DOM очень полезна для приложений, которым требуется повторный произвольный доступ к разным частям документа.

Простой API для XML (SAX)

Члены списка рассылки XML-DEV совместно разработали простой API для XML (SAX) в первую очередь для того, чтобы обеспечить облегченную альтернативу DOM. В противоположность DOM SAX предоставляет событийную модель для обработки XML-документов. Вместо формирования полного образа документа в памяти, SAX-анализатор использует механизм обратного вызова (callback) для информирования клиентского приложения о нахождении различных синтаксических конструкций XML. Анализатор не хранит какой-либо информации о состоянии и потому может обрабатывать очень большие документы, не требуя больших объемов памяти.