



# Содержание

Предисловие .....	9
О структуре книги .....	11
Типографские соглашения .....	13
О примерах кода .....	14
Благодарности .....	15
<b>Глава 1. Язык XPath .....</b>	<b>17</b>
1.1. Применение осей .....	18
1.2. Фильтрация узлов .....	24
1.3. Работа с последовательностями .....	27
1.4. Включение условий в выражения if .....	29
1.5. Исключение рекурсии с помощью выражений for .....	32
1.6. Упрощение сложной логики с помощью кванторов .....	34
1.7. Операции над множествами .....	36
1.8. Сравнение узлов .....	37
1.9. Как ужиться с расширенной системой типов в XPath 2.0 .....	38
1.10. Как воспользоваться расширенной системой типов в XPath 2.0 .....	41
<b>Глава 2. Строки .....</b>	<b>43</b>
2.0. Введение .....	43
2.1. Завершается ли данная строка указанной подстрокой? .....	44
2.2. Нахождение позиции начала подстроки .....	44
2.3. Удаление заданных символов из строки .....	45
2.4. Поиск подстроки с конца строки .....	49
2.5. Повторение строки N раз .....	53
2.6. Обращение строки .....	57
2.7. Замена текста .....	61
2.8. Преобразование регистра .....	67
2.9. Разбиение строки на лексемы .....	70
2.10. Как обойтись без регулярных выражений .....	73
2.11. Использование регулярных выражений .....	74
2.12. Расширения EXSLT для работы со строками .....	76
<b>Глава 3. Математические операции над числами .....</b>	<b>80</b>
3.0. Введение .....	80
3.1. Форматирование чисел .....	81
3.2. Округление чисел с заданной точностью .....	90

3.3. Преобразование римских числительных в числа .....	93
3.4. Преобразование из одной системы счисления в другую .....	96
3.5. Реализация стандартных математических функций .....	100
3.6. Вычисление сумм и произведений .....	114
3.7. Нахождение минимума и максимума .....	120
3.8. Вычисление статистических функций .....	128
3.9. Вычисление комбинаторных функций .....	132
3.10. Проверка битов .....	134
<b>Глава 4. Даты и время .....</b>	<b>141</b>
4.0. Введение .....	141
4.1. Вычисление дня недели .....	143
4.2. Вычисление последнего дня месяца .....	145
4.3. Получение названий дней и месяцев .....	146
4.4. Вычисление юлианского и абсолютного дня, соответствующих заданной дате .....	151
4.5. Вычисление номера недели, соответствующего заданной дате .....	156
4.6. Юлианский календарь .....	157
4.7. Календарь ISO .....	158
4.8. Исламский календарь .....	161
4.9. Еврейский календарь .....	164
4.10. Форматирование даты и времени .....	174
4.11. Определение светских и церковных праздников .....	187
<b>Глава 5. Отбор и обход .....</b>	<b>191</b>
5.0. Введение .....	192
5.1. Игнорирование элементов-дубликатов .....	194
5.2. Отбор всех элементов, кроме одного .....	200
5.3. Отбор узлов по контексту .....	201
5.4. Выполнение обхода в прямом порядке .....	203
5.5. Выполнение обхода в обратном порядке .....	208
5.6. Выполнение обхода во внутреннем порядке .....	210
5.7. Выполнение обхода по уровням .....	214
5.8. Обработка узлов по позиции .....	221
<b>Глава 6. XSLT 2.0 .....</b>	<b>229</b>
6.0. Введение .....	229
6.1. Преобразование простых именованных шаблонов в функции XSLT .....	229
6.2. Для группировки пользуйтесь командой for-each-group, а не методом Мюнха .....	231

6.3. Модульность и режимы .....	235
6.4. Применение типов в целях безопасности и точности выражения намерений .....	236
6.5. Избегайте подводных камней при переносе с XSLT 1.0 на 2.0 .....	238
6.6. Эмуляция объектно-ориентированных методик повторного использования и паттернов проектирования .....	240
6.7. Обработка неструктурированного текста с помощью регулярных выражений .....	245
6.8. Решение трудных задач сериализации с помощью таблиц символов .....	249
6.9. Вывод нескольких документов .....	251
6.10. Обработка строковых литералов, содержащих кавычки .....	253
6.11. Новые возможности старых конструкций XSLT 1.0 .....	253
<b>Глава 7. Преобразование XML в текст .....</b>	<b>260</b>
7.0. Введение .....	260
7.1. Пустое пространство .....	261
7.2. Экспорт XML в файл с разделителями полей .....	267
7.3. Создание отчета с несколькими колонками .....	287
7.4. Отображение иерархии .....	299
7.5. Вывод текста с нумерацией .....	308
7.6. Вывод текста в области заданной ширины с заданным выравниванием .....	319
<b>Глава 8. Преобразование XML в XML .....</b>	<b>323</b>
8.0. Введение .....	323
8.1. Преобразование атрибутов в элементы .....	324
8.2. Преобразование элементов в атрибуты .....	327
8.3. Переименование элементов или атрибутов .....	332
8.4. Объединение документов с одной и той же схемой .....	339
8.5. Объединение документов с различными схемами .....	345
8.6. Расщепление документов .....	351
8.7. Уплотнение иерархии XML .....	353
8.8. Углубление иерархии XML .....	357
8.9. Реорганизация иерархии XML .....	363
<b>Глава 9. Опрос XML-документа .....</b>	<b>368</b>
9.0. Введение .....	369
9.1. Выполнение теоретико-множественных операций над наборами узлов .....	370

9.2. Выполнение теоретико-множественных операций над наборами узлов с использованием семантики значений .....	373
9.3. Сравнение наборов узлов на равенство по значению .....	385
9.4. Выполнение запросов, сохраняющих структуру .....	390
9.5. Соединения .....	393
9.6. Реализация на XSLT сценариев, приведенных в спецификации W3C XML Query .....	398
<b>Глава 10. Преобразование XML в HTML .....</b>	<b>433</b>
10.0. Введение .....	433
10.1. Использование XSLT в качестве языка стилизации .....	434
10.2. Создание документов, связанных гиперссылками .....	443
10.3. Создание HTML-таблиц .....	446
10.4. Создание фреймов .....	455
10.5. Создание таблиц стилей, управляемых данными .....	461
10.6. Создание замкнутого преобразования .....	468
10.7. Заполнение формы .....	473
<b>Глава 11. Преобразование XML в SVG .....</b>	<b>480</b>
11.0. Введение .....	480
11.1. Преобразование имеющейся заготовки SVG .....	482
11.2. Создание повторно используемых утилит генерации SVG для графиков и диаграмм .....	490
11.3. Создание графического представления деревьев .....	532
11.4. Создание интерактивных Web-страниц, включающих SVG .....	543
<b>Глава 12. Генерация кода .....</b>	<b>554</b>
12.0. Введение .....	555
12.1. Генерация определений констант .....	565
12.2. Генерация предложения switch .....	570
12.3. Генерация кода заглушки обработчика сообщения .....	575
12.4. Генерация оберток для данных .....	579
12.5. Генерация функций форматированной печати .....	584
12.6. Генерация Web-клиента для тестирования ввода данных .....	593
12.7. Генерация CGI-сценария для обработки тестовых данных .....	595
12.8. Генерация кода из UML-моделей, описанных на языке XMI .....	601
12.9. Генерация XSLT из XSLT .....	620

<b>Глава 13. Рецепты применения XSLT в вертикальных приложениях</b> .....	625
13.0. Введение .....	625
13.1. Преобразование документов из формата Visio VDX в формат SVG .....	626
13.2. Работа с электронными таблицами в формате Excel XML .....	641
13.3. Генерация тематических карт из UML-модели с помощью XMI .....	653
13.4. Генерация Web-сайтов из тематических карт .....	673
13.5. Генерация документации о SOAP из WSDL-документа .....	689
<b>Глава 14. Расширение и встраивание XSLT</b> .....	706
14.0. Введение .....	706
14.1. Функции расширения в Saxon .....	707
14.2. Элементы расширения в Saxon .....	708
14.3. Функции расширения в Xalan-Java 2 .....	709
14.4. Описание Java-функций расширения в формате класса .....	709
14.5. Описание Java-функций расширения в формате пакета .....	710
14.6. Описание Java-функций расширения в формате Java .....	710
14.7. Функции расширения на языке сценариев с использованием встроенного сценария .....	711
14.8. Элементы расширения в Xalan-Java 2 .....	711
14.9. Реализация элемента расширения на языке Java .....	711
14.10. Реализация элемента расширения на сценарном языке .....	712
14.11. Функции расширения в MSXML .....	713
14.12. Использование встроенных расширений Saxon и Xalan .....	714
14.13. Расширение XSLT с помощью JavaScript .....	730
14.14. Реализация функций расширения на языке Java .....	736
14.15. Реализация элементов расширения на языке Java .....	744
14.16. Работа с XSLT в программах на языке Perl .....	760
14.17. Работа с XSLT в программах на языке Java .....	763
<b>Глава 15. Тестирование и отладка</b> .....	767
15.0. Введение .....	767
15.1. Эффективное использование xsl:message .....	768
15.2. Трассировка потока обработки документа таблицей стилей .....	771

15.3. Автоматизация вставки отладочной печати .....	778
15.4. Встраивание тестовых данных в служебные таблицы стилей .....	786
15.5. Организация автономных тестов .....	791
15.6. Тестирование граничных условий и ошибочных данных .....	794
<b>Глава 16. Обобщенное и функциональное программирование .....</b>	<b>798</b>
16.0. Введение .....	798
16.1. Создание полиморфного XSLT-кода .....	805
16.2. Создание обобщенных функций агрегирования элементов .....	814
16.3. Создание обобщенных функций ограниченного агрегирования .....	828
16.4. Создание обобщенных функций отображения .....	836
16.5. Создание обобщенных генераторов наборов узлов .....	846
<b>Алфавитный указатель .....</b>	<b>853</b>



## Предисловие

XSLT (Extensible Stylesheet Language Transformations) – это технология преобразования XML-документов в другие форматы. Некоторые считают, что освоить ее трудно. Однако тот факт, что в основе технологии лежат шаблоны, делают ее прекрасным кандидатом для изучения на примерах, особенно, если учесть что имеющийся образец зачастую легко приспособить для других целей. Благодаря спецификации XSLT 2.0 выразительность и элегантность XSLT заметно возросли, но язык при этом стал сложнее.

Когда я только начинал работать с XSLT (и позднее, когда приступил к изучению XSLT 2.0), мне очень не хватало сборника рецептов, из которого я мог бы черпать готовые решения стоящих передо мной задач. Первой книгой такого рода, которая попала мне на глаза, была *Perl Cookbook*, выпущенная издательством O'Reilly. Она заставила меня сначала преодолеть нежелание изучать язык Perl, а затем и полюбить его, в чем оригинальная книга Ларри Уолла (*Programming Perl* – знаменитая «книга с верблюдом») не преуспела. Мне кажется, что такие сборники рецептов нужны, поскольку многим разработчикам недостаточно просто заставить программу работать; они хотели бы в совершенстве овладеть технологией и применять хорошо зарекомендовавшие себя приемы. При этом получить необходимую информацию желательно быстро. А нет лучшего способа глубоко изучить предмет, чем воспользоваться опытом тех, кто уже нашел удачный подход к проблеме.

Острое желание иметь подобную книгу вскоре переросло в желание самому написать ее, тем более что у меня уже подобралось несколько полезных рецептов; что-то придумали другие, а что-то – я сам. Однако я не хотел просто переписывать учебник по XSLT, по-другому сгруппировав материал. Я поставил целью создать полезный ресурс, в котором освещались бы некоторые неочевидные способы применения этой технологии. Заодно я надеялся привлечь внимание тех разработчиков, которые еще не заинтересовались XSLT и тем самым, на мой взгляд, упустили из виду один из наиболее эффективных инструментов работы с XML-документами. Если вы принадлежите к их числу, прошу вас прочитать хотя бы еще несколько абзацев, где я расскажу, чем ценен XSLT и как эта книга поможет вам осознать его потенциал.

XSLT – это язык, который находится одновременно в центре и на обочине магистрального пути развития технологий разработки программного обеспечения. Когда я работал над первым изданием этой книги, мне часто приходилось объяснять знакомым, что такое XSLT и почему так важно тратить время на написание целой книги о нем. Мои знакомые были наслышаны о Java, Perl и даже XML, но ничего не знали об XSLT. Я также видел, как растет потребность в консультациях

по XSLT, в списках рассылки и большем внимании индустрии к этой технологии, которое выражалось бы в книгах, статьях и развитых инструментах поддержки. Хотя число пользователей XSLT возрастает с каждым днем, многие профессионалы все еще не понимают, что это такое и с чем его едят. Я надеюсь, что по мере появления реализаций XSLT 2.0 эта технология распространится шире, однако уверенности в этом нет, в частности из-за конкуренции со стороны XQuery 1.0 и других способов манипулирования XML-документами. Но одно можно сказать точно: изучение XSLT 2.0 не станет потерянным впустую временем, поскольку область его применения будет расширяться, хотя, быть может, и не лавинообразно. К тому же знакомство с XSLT углубит ваши представления о том, как можно обрабатывать XML, даже если в конечном итоге вы остановитесь на другой технологии.

Хотя XSLT 1.0 – вполне зрелый язык и спецификация XSLT 2.0 была принята не так уж давно, у меня есть ощущение, что больше половины людей и компаний, работающих с XML, не используют XSLT. Сравнительно недавно один мой знакомый, который в курсе всех новейших технологий, говорил мне, что XSLT – это просто еще один язык стилизации. Такое недопонимание простительно, поскольку об XSLT судят по первым трем буквам аббревиатуры (XSL – расширенный язык таблиц стилей) и по директиве, с которой начинаются большинство XSLT-программ (`xsl:stylesheet`). Но самой важной является как раз последняя буква T, означающая *Transformations* (преобразования). Именно это делает язык таким ценным, и по этой причине я им и заинтересовался. При написании этой книги я, в частности, намеревался показать, что XSLT применим к решению самых разных задач. Кроме того, я хотел, чтобы эта книга стала для пользователей начального и среднего уровня тем местом, где можно найти большинство распространенных приемов работы с XSLT. И, наконец, я хотел продемонстрировать, что вообще можно делать с XSLT и тем самым побудить тех, кто уже знает этот язык, к более глубокому изучению, а остальных – присоединиться к компании «преобразователей XML».

Мне доводилось слышать самые разные мнения о том, что такое информатика. Высказывания типа «любое вычисление – это просто более или менее хитроумное манипулирование битами», «компьютеры – не более чем очень усложненные перемальватели чисел» или «все, что может делать компьютер, возможно выразить в терминах манипулирования символами» в какой-то степени верны. Но я бы хотел предложить и свое собственное обобщение: «Любая задача, решаемая с помощью программы, может быть описана в терминах преобразований». Тот, кто овладел искусством преобразований, овладел и информатикой. Преобразованиями занимается процессор, преобразования лежат в основе алгоритмов, преобразования – это суть работы программистов. И именно преобразования составляют смысл языка XSLT, по крайней мере, когда на вход подается XML-документ (а иногда и нечто иное). Разумеется, XSLT – не единственный язык преобразований, и, как и для тысяч предшествовавших ему языков, неясно, будет ли он развиваться независимо или окажется поглощенной следующей «потрясающей инновацией». Очевидно лишь, что лежащие в основе XSLT идеи никуда не денутся,



поскольку они стары, как и сама информатика. Эта книга поможет читателю понять эти идеи и научиться применять их к конкретным задачам.

## О структуре книги

Чтобы сделать книгу полезной максимально широкому кругу читателей, я сохранил большую часть примеров XSLT 1.0 из первого издания. Но добавил и решения на базе XSLT 2.0 в тех случаях, когда они оказывались проще или элегантнее. Иногда я также привожу «2.0-решения» задач, которые было бы очень трудно или даже невозможно решить в рамках версии 1.0. Примеры для версий 1.0 и 2.0 приводятся в разных подразделах. Надеюсь, что так читателям будет проще найти то, что их интересует. Во многих примерах я не даю отдельного решения для версии 2.0. Как правило, это объясняется тем, что, на мой взгляд, «1.0-решение» будет работать и в версии 2.0, а специальная адаптация под 2.0 мало что дает. Искренне надеюсь, что мое желание сохранить деревья и сэкономить время не вызовет раздражения у читателя.

Версии XSLT 1.0 и 2.0 опираются на фундамент, заложенный спецификациями XPath 1.0 и 2.0 соответственно. Некоторые читатели первого издания пеняли мне на то, что я не уделил должного внимания XPath. Глава 1 написана отчасти для того, чтобы удовлетворить их, а отчасти потому, что язык XPath 2.0 заметно расширился и усложнился.

Одно из самых простых преобразований – это обработка последовательности символов, то есть *строки*. В отличие от древнего языка SNOBOL и относительно современного Perl, XSLT не проектировался специально для манипулирования строками. Однако в главе 2 показано, что практически все, что необходимо при работе со строками, можно выполнить, оставаясь в рамках XSLT, причем новые функции, появившиеся в XSLT 2.0, значительно упрощают задачу.

Еще один вид низкоуровневых преобразований – это численные преобразования (собственно, это не что иное, как *математика*). Они пронизывают программирование снизу доверху просто потому, что измерения и расчеты – неотъемлемая часть самой жизни. В главе 3 показано, как обогатить «математические способности» XSLT, хотя этот язык и не задумывался как полноценная замена Фортрану.

Манипулирование датами и временем заложено в самой природе человека, интерес к часам, календарям и точному прогнозированию часто становился движущей силой прогресса. Глава 4 содержит рецепты работы с датами и временем, которые восполняют недостатки, присущие стандартному XSLT 1.0. Там же описываются долгожданные функции, которые наконец-то были добавлены в XSLT 2.0. Рассматривая в этой главе интригующие и трудные задачи, связанные с преобразованием дат, мы приведем как готовые решения, так и ссылки на ресурсы, относящиеся к летоисчислению.

Для любого преобразования нужно прежде всего указать объект, к которому оно применяется. Если этот объект составной, то в ходе преобразования нужно будет обойти его части. Этой теме посвящена глава 5, в которой исследуются

проблемы, для решения которых и предназначен XSLT. Здесь мы представляем XML-документ в виде дерева и показываем, как XSLT позволяет манипулировать такими деревьями. Попутно приводятся советы по поводу достижения максимальной производительности при обработке XML-документов.

Глава 6 была заново написана для второго издания. Она целиком посвящена версии XSLT 2.0. Читателям, которых больше всего интересуют новшества, появившиеся в этой версии, рекомендуется сначала прочесть главы 1 и 6, а затем для более углубленного понимания познакомиться с примерами ее применения, рассыпанными по всему тексту.

Задолго до появления различных текстовых процессоров, HTML и PDF существовал старый добрый текст. Весьма важна задача преобразования данных из формата, предназначенного для обработки компьютером, к виду, удобному человеку. Если исходные данные представлены в формате XML, то для ее решения XSLT подходит идеально. В главе 7 демонстрируется, как из XML-документа можно извлечь текст, пригодный для вывода на терминал, подачи на вход текстового редактора или для импорта в программы, требующие, чтобы значения были как-то отделены друг от друга, например, запятыми.

XML быстро становится универсальным форматом для передачи информации, и есть все признаки того, что эта тенденция будет и дальше набирать силу, а не сойдет на нет. Поэтому очень часто одни XML-документы приходится преобразовывать в другие. В главе 8 рассматриваются именно такие преобразования. Здесь показано, как XML-документы можно разбивать на части, объединять, преобразовывать из иерархического вида в плоский, очищать и выполнять иные трансформации с помощью сравнительно небольших XSLT-шаблонов.

Многие преобразования сводятся просто к извлечению информации из имеющихся данных для получения ответа на тот или иной вопрос. Глава 9 – это сокровищница рецептов, иллюстрирующих применение XSLT в качестве языка запросов. В ней приведено немало примеров, моделирующих запросы, которые вполне могут возникнуть и у вас.

Еще одна важная сфера применения XSLT – это HTML-разметка. В главе 10 показано, как решать задачи генерирования Web-контента, в том числе ссылок, таблиц, фреймов, форм и выполнять другие преобразования на стороне клиента.

Задача графического программирования – преобразование данных к визуально воспринимаемому виду. Не стоит думать, что XSLT – это язык графического программирования. Однако, если нужно выполнить преобразование в формат SVG (Scalable Vector Graphics), то XSLT поможет добиться впечатляющих результатов. В главе 11 демонстрируется, как преобразовать исходные данные в столбчатые или секторные диаграммы, линейные графики и другие графические компоненты. Здесь же рассматриваются преобразования XML в древовидное представление. В этой главе подчеркивается, что преобразования – это конструкции, которые можно комбинировать для получения самых разных результатов на выходе.

Меня всегда интересовала задача автоматической генерации кода. С этим преобразованием люди пока справляются лучше компьютеров (к счастью для тех, кто зарабатывает на жизнь программированием). Однако иногда лучше написать