

LINQ

Язык интегрированных
запросов в C# 2010

ДЛЯ ПРОФЕССИОНАЛОВ

Pro
LINQ
Language Integrated
Query in C# 2010

Adam Freeman and
Joseph C. Rattz, Jr.

Apress®

LINQ

Язык интегрированных
запросов в C# 2010

ДЛЯ ПРОФЕССИОНАЛОВ

Адам Фримен
Джозеф Раттц-мл.



Москва • Санкт-Петербург • Киев
2011

ББК 32.973.26-018.2.75
Ф88
УДК 681.3.07

Издательский дом “Вильямс”
Зав. редакцией *С.Н. Тригуб*
Перевод с английского *Н.А. Мухина*
Под редакцией *Ю.Н. Артеменко*

По общим вопросам обращайтесь в Издательский дом “Вильямс” по адресу:
info@williamspublishing.com, <http://www.williamspublishing.com>

Фримен, Адам, **Ратц-мл.**, Джозеф С.

Ф88 LINQ: язык интегрированных запросов в C# 2010 для профессионалов. : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2011. — 656 с. : ил. — Парал. тит. англ.

ISBN 978-5-8459-1701-0 (рус.)

ББК 32.973.26-018.2.75

Все названия программных продуктов являются зарегистрированными торговыми марками соответствующих фирм.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на это нет письменного разрешения издательства APress, Berkeley, CA.

Authorized translation from the English language edition published by APress, Copyright © 2010 by Adam Freeman and Joseph C. Rattz, Jr.

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

Russian language edition is published by Williams Publishing House according to the Agreement with R&I Enterprises International, Copyright © 2011.

Научно-популярное издание

Адам Фримен, Джозеф С. Ратц-мл.

LINQ: язык интегрированных запросов в C# 2010 для профессионалов

Верстка *Т.Н. Артеменко*
Художественный редактор *В.Г. Павлютин*

Подписано в печать 24.01.2011. Формат 70×100/16.
Гарнитура Times. Печать офсетная.
Усл. печ. л. 52,89. Уч.-изд. л. 37,6.
Тираж 1500 экз. Заказ № 0000.

Отпечатано по технологии CtP
в ОАО “Печатный двор” им. А. М. Горького
197110, Санкт-Петербург, Чкаловский пр., 15.

ООО “И. Д. Вильямс”, 127055, г. Москва, ул. Лесная, д. 43, стр. 1

ISBN 978-5-8459-1701-0 (рус.)
ISBN 978-1430-22653-6 (англ.)

© Издательский дом “Вильямс”, 2011
© by Adam Freeman and Joseph C. Rattz, Jr., 2010

Оглавление

Часть I. LINQ: язык интегрированных запросов в C# 2010	17
Глава 1. Знакомство с LINQ	18
Глава 2. Расширения языка C# для LINQ	32
Часть II. LINQ to Objects	61
Глава 3. Введение в LINQ to Objects	62
Глава 4. Отложенные операции	71
Глава 5. Не отложенные операции	135
Часть III. LINQ to XML	181
Глава 6. Введение в LINQ to XML	182
Глава 7. Интерфейс LINQ to XML API	187
Глава 8. Операции LINQ to XML	260
Глава 9. Дополнительные возможности LINQ to XML	285
Часть IV. LINQ to DataSet	319
Глава 10. Операции LINQ to DataSet	320
Глава 11. Дополнительные возможности LINQ to DataSet	350
Часть V. LINQ to SQL	357
Глава 12. Введение в LINQ to SQL	358
Глава 13. Советы и инструменты, связанные с LINQ to SQL	368
Глава 14. Операции LINQ to SQL для баз данных	389
Глава 15. Сущностные классы LINQ to SQL	428
Глава 16. Класс DataContext в LINQ to SQL	469
Глава 17. Конфликты параллельного доступа LINQ to SQL	522
Глава 18. Дополнительные возможности LINQ to SQL	539
Часть VI. LINQ to Entities	549
Глава 19. Введение в LINQ to Entities	550
Глава 20. Операции LINQ to Entities	559
Глава 21. Классы LINQ to Entities	593
Часть VII. Parallel LINQ	613
Глава 22. Введение в Parallel LINQ	614
Глава 23. Использование Parallel LINQ	619
Глава 24. Операции Parallel LINQ	631
Предметный указатель	647

Содержание

Об авторах	15
О техническом рецензенте	15
Благодарности	16
От издательства	16
Исходный код примеров	16
Часть I. LINQ: язык интегрированных запросов в C# 2010	17
Глава 1. Знакомство с LINQ	18
Сдвиг парадигмы	18
Запрос к XML	19
Запрос к базе данных SQL Server	20
Появление LINQ	21
LINQ поддерживает запросы данных	21
Как получить LINQ	22
LINQ предназначен не только для запросов	23
Советы начинающим	25
Используйте ключевое слово var, когда запутались	26
Используйте операции Cast или OfType для унаследованных коллекций	27
Отдавайте предпочтение операции OfType перед Cast	28
Не рассчитывайте на безошибочность запросов	28
Используйте преимущество отложенных запросов	29
Используйте свойство Log из DataContext	30
Используйте форум LINQ	31
Резюме	31
Глава 2. Расширения языка C# для LINQ	32
Дополнения языка C#	32
Лямбда-выражения	32
Деревья выражений	37
Ключевое слово var, инициализация объектов и анонимные типы	38
Расширяющие методы	43
Частичные методы	47
Выражения запросов	49
Резюме	58
Часть II. LINQ to Objects	61
Глава 3. Введение в LINQ to Objects	62
Обзор LINQ to Objects	62
Интерфейс IEnumerable<T>, последовательности и стандартные операции запросов	63
Возврат IEnumerable<T>, выдача и отложенные запросы	64
Делегаты Func	67
Алфавитный указатель стандартных операций запросов	68
История о двух синтаксисах	69
Резюме	70

Глава 4. Отложенные операции	71
Необходимые пространства имен	71
Необходимые сборки	71
Общие классы	71
Организация отложенных операций по назначению	73
Ограничение	73
Проекция	75
Разбиение	83
Конкатенация	90
Упорядочивание	92
Соединение	107
Группирование	111
Множества	116
Преобразование	121
Элемент	127
Генерация	131
Резюме	134
Глава 5. Не отложенные операции	135
Необходимые пространства имен	135
Общие классы	135
Организация не отложенных операций по назначению	138
Преобразование	138
Эквивалентность	149
Элемент	151
Квантификаторы	163
Агрегация	168
Резюме	180
Часть III. LINQ to XML	181
Глава 6. Введение в LINQ to XML	182
Введение	184
Обман W3C DOM XML API	184
Резюме	186
Глава 7. Интерфейс LINQ to XML API	187
Необходимые пространства имен	187
Значимые проектные усовершенствования API-интерфейса	187
Конструирование деревьев XML было упрощено с помощью функционального конструирования	188
Центральная роль элемента вместо документа	190
Имена, пространства имен и префиксы	191
Извлечение значения узла	194
Объектная модель LINQ to XML	196
Отложенное выполнение запросов, удаление узлов и “проблема Хэллоуина”	197
Создание XML	199
Создание элементов с помощью XElement	200
Создание атрибутов с помощью XAttribute	202
Создание комментариев с помощью XComment	203

8 Содержание

Создание контейнеров с помощью XContainer	203
Создание объявлений с помощью XDeclaration	204
Создание типов документов с помощью XDocumentType	204
Создание документов с помощью XDocument	206
Создание имен с помощью XName	206
Создание пространств имен с помощью XNamespace	207
Создание узлов с помощью XNode	207
Создание инструкций обработки с помощью XProcessingInstruction	207
Создание потоковых элементов с помощью XStreamingElement	209
Создание текста с помощью XText	210
Создание CDData с помощью XCData	211
Вывод XML	211
Сохранение с помощью XDocument.Save()	211
Сохранение с помощью XElement.Save()	212
Ввод XML	213
Загрузка с помощью XDocument.Load()	213
Загрузка с помощью XElement.Load()	215
Разбор содержимого с помощью методов XDocument.Parse() или XElement.Parse()	215
Обход XML	216
Свойства обхода	217
Обход вперед с помощью XNode.NextNode	217
Обход назад с помощью XNode.PreviousNode	218
Методы обхода	220
Модификация XML	232
Добавление узлов	233
Удаление узлов	237
Обновление узлов	239
Вызов XElement.SetElementValue() на дочерних объектах XElement	242
Атрибуты XML	244
Создание атрибута	244
Обход атрибутов	244
Модификация атрибутов	247
Аннотации XML	251
Добавление аннотаций с помощью XObject.AddAnnotation()	251
Обращение к аннотациям с помощью XObject.Annotation() или XObject.Annotations()	251
Удаление аннотаций с помощью XObject.RemoveAnnotations()	251
Пример аннотаций	251
События XML	254
XObject.Changing	255
XObject.Changed	255
Несколько примеров событий	255
Трюк, забава или неопределенность?	258
Резюме	259
Глава 8. Операции LINQ to XML	260
Введение в операции LINQ to XML	260
Ancestors	261
Прототипы	261
Примеры	261

AncestorsAndSelf	265
Прототипы	265
Примеры	265
Attributes	267
Прототипы	267
Примеры	267
DescendantNodes	269
Прототипы	269
Примеры	269
DescendantNodesAndSelf	270
Прототипы	271
Примеры	271
Descendants	272
Прототипы	272
Примеры	272
DescendantsAndSelf	274
Прототипы	274
Примеры	274
Elements	276
Прототипы	276
Примеры	277
InDocumentOrder	278
Прототипы	278
Примеры	279
Nodes	280
Прототипы	280
Примеры	280
Remove	281
Прототипы	281
Примеры	282
Резюме	284
Глава 9. Дополнительные возможности LINQ to XML	285
Необходимые пространства имен	285
Запросы	286
Отсутствие иерархического спуска	286
Сложный запрос	288
Трансформации	293
Трансформации с использованием XSLT	294
Трансформация с использованием функционального конструирования	295
Советы	298
Проверка достоверности	303
Расширяющие методы	303
Прототипы	303
Получение схемы XML	304
Примеры	306
XPath	316
Прототипы	316
Примеры	316
Резюме	317

10 Содержание

Часть IV. LINQ to DataSet	319
Глава 10. Операции LINQ to DataSet	320
Необходимые сборки	321
Необходимые пространства имен	321
Общий код для примеров	321
Операции множеств DataRow	322
Distinct	323
Except	326
Intersect	328
Union	330
SequenceEqual	331
Операции над полями DataRow	333
Field<T>	336
SetField<T>	341
Операции DataTable	343
AsEnumerable	344
CopyToDataTable<DataRow>	344
Примеры	345
Резюме	349
Глава 11. Дополнительные возможности LINQ to DataSet	350
Необходимые пространства имен	350
Типизированные DataSet	350
Собираем все вместе	352
Резюме	355
Часть V. LINQ to SQL	357
Глава 12. Введение в LINQ to SQL	358
Введение в LINQ to SQL	359
DataContext	360
Сущностные классы	361
Ассоциации	361
Обнаружение конфликтов параллельного доступа	362
Разрешение конфликтов параллельного доступа	362
Предварительные условия для запуска примеров	363
Получение соответствующей версии базы данных Northwind	363
Генерация сущностных классов Northwind	363
Генерация XML-файла отображения Northwind	364
Использование LINQ to SQL	364
IQueryable<T>	365
Некоторые общие методы	365
GetStringFromDb()	365
ExecuteStatementInDb()	366
Резюме	367
Глава 13. Советы и инструменты, связанные с LINQ to SQL	368
Введение	368
Советы	368
Используйте свойство DataContext.Log	369

Используйте метод <code>GetChangeSet()</code>	370
Подумайте об использовании частичных классов или файлов отображения	370
Подумайте об использовании частичных методов	370
Инструменты	370
SQLMetal	370
Object Relational Designer	376
Совместное использование SQLMetal и Object Relational Designer	387
Резюме	388
Глава 14. Операции LINQ to SQL для баз данных	389
Предварительные условия для запуска примеров	389
Некоторые общие методы	389
Использование API-интерфейса LINQ to SQL	390
Стандартные операции для баз данных	390
Вставки	390
Запросы	393
Обновления	415
Удаления	418
Переопределение операторов модификации базы данных	421
Переопределение метода <code>Insert</code>	421
Переопределение метода <code>Update</code>	421
Переопределение метода <code>Delete</code>	422
Пример	422
Переопределение в Object Relational Designer	424
Соображения	424
Трансляция SQL	424
Резюме	426
Глава 15. Сущностные классы LINQ to SQL	428
Предварительные условия для запуска примеров	428
Сущностные классы	428
Создание сущностных классов	428
XML-схема внешнего файла отображения	456
Сравнение проекций на сущностные и на несущностные классы	456
Расширение сущностных классов с помощью частичных методов	461
Важные классы API-интерфейса <code>System.Data.Linq</code>	462
<code>EntitySet<T></code>	463
<code>EntityRef<T></code>	463
<code>Table<T></code>	465
<code>IExecuteResult</code>	465
<code>ISingleResult<T></code>	466
<code>IMultipleResults</code>	467
Резюме	468
Глава 16. Класс <code>DataContext</code> в LINQ to SQL	469
Предварительные условия для запуска примеров	469
Некоторые общие методы	469
Использование API-интерфейса LINQ to SQL	469
Класс <code>[Your]DataContext</code>	469
Класс <code>DataContext</code>	470
Класс <code>DataContext</code> реализует интерфейс <code>IDisposable</code>	472

12 Содержание

Основное назначение	473
Время жизни контекста данных	478
DataContext() и [Your]DataContext()	479
SubmitChanges()	490
DatabaseExists()	496
CreateDatabase()	496
DeleteDatabase()	498
CreateMethodCallQuery()	498
ExecuteQuery()	500
Translate()	502
ExecuteCommand()	503
ExecuteMethodCall()	505
GetCommand()	511
GetChangeSet()	512
GetTable()	514
Refresh()	515
Резюме	521
Глава 17. Конфликты параллельного доступа LINQ to SQL	522
Предварительные условия для запуска примеров	522
Некоторые общие методы	522
Использование API-интерфейса LINQ to SQL	522
Конфликты параллелизма	522
Оптимистический параллелизм	523
Пессимистический параллелизм	533
Альтернативный подход для средних звеньев и серверов	535
Резюме	538
Глава 18. Дополнительные возможности LINQ to SQL	539
Предварительные условия для запуска примеров	539
Использование API-интерфейса LINQ to SQL	539
Использование API-интерфейса LINQ to XML	539
Представления базы данных	539
Наследование сущностных классов	541
Транзакции	546
Резюме	548
Часть VI. LINQ to Entities	549
Глава 19. Введение в LINQ to Entities	550
Введение	551
ObjectContext	552
Сущностные классы	552
Ассоциации	553
Предварительные условия для запуска примеров	553
Получение соответствующей версии базы данных Northwind	553
Генерация сущностной модели данных Northwind	553
Использование API-интерфейса LINQ to Entities	556
IQueryable<T>	556
Некоторые общие методы	556
GetStringFromDo()	557

ExecuteStatementInDb()	557
Резюме	558
Глава 20. Операции LINQ to Entities	559
Предварительные условия для запуска примеров	559
Некоторые общие методы	559
Стандартные операции базы данных	559
Вставки	560
Запросы	565
Обновления	578
Удаления	579
Удаление связанных объектов	581
Управление параллельным доступом	586
Включение проверок параллелизма	587
Обработка конфликтов параллелизма	588
Резюме	592
Глава 21. Классы LINQ to Entities	593
Предварительные условия для запуска примеров	593
КлассObjectContext	593
Конструктор	594
DatabaseExists()	595
DeleteDatabase()	596
CreateDatabase()	596
SaveChanges()	596
Refresh()	597
AddObject()	598
CreateObject()	599
DeleteObject()	600
EntityObject	600
Конструктор	600
Фабричный метод	601
Примитивные свойства	603
Навигационные свойства	604
EntityReference	606
Load()	606
Value	607
EntityCollection	607
Add()	607
Remove()	609
Clear()	610
Contains()	610
Load()	611
Count	611
Резюме	612
Часть VII. Parallel LINQ	613
Глава 22. Введение в Parallel LINQ	614
Введение	614
Parallel LINQ предназначен для объектов	617

14 Содержание

Использование API-интерфейса Parallel LINQ	618
Резюме	618
Глава 23. Использование Parallel LINQ	619
Создание запроса Parallel LINQ	619
Предохранение порядка результатов	621
Управление параллелизмом	624
Принудительное параллельное выполнение	624
Ограничение степени параллелизма	625
Обработка исключений	625
Запросы без результатов	628
Создание диапазонов и повторов	630
Резюме	630
Глава 24. Операции Parallel LINQ	631
Операции создания ParallelQuery	631
AsParallel	631
Range	634
Repeat	635
Empty	636
Операции управления выполнением	636
AsOrdered	636
AsUnordered	638
AsSequential	639
AsEnumerable	640
WithDegreeOfParallelism	641
Прототипы	641
WithExecutionMode	641
WithMergeOptions	642
Операции преобразования	644
Cast	644
OfType	645
Операция ForAll	646
Прототипы	646
Примеры	646
Резюме	646
Предметный указатель	647

Об авторах

Адам Фримен — профессионал в области информационных технологий, который занимал ведущие должности в ряде компаний, последняя из которых — директор по развитию технологий и главный операционной директор в глобальном банке. Он написал несколько книг по Java и .NET и давно интересуется параллельными вычислениями.

Джозеф Раттц-мл. неосознанно начал свою карьеру в разработке программного обеспечения в 1990 г., когда друг попросил его помочь в написании текстового редактора ANSI под названием ANSI Master для компьютера Commodore Amiga. Вскоре за этим последовала игра в палач (The Gallows — “Виселица”). От этих программ, написанных на компилируемом Basic, он перешел к программированию на C, в поисках более высокой скорости и мощности. После этого Джо разрабатывал приложения для *JumpDisk* — журнала Amiga на дисках, а также для журнала *Amiga World*. Из-за того, что ему пришлось работать в маленьком городе в относительной изоляции, Джо изучил все неправильные способы написания кода. Это обучение происходило при попытках усовершенствовать плохо написанные приложения, в процессе которых он осознал важность написания хорошо сопровождаемого кода. Впервые познакомившись с отладчиком уровня исходного кода, он влюбился в него с первого взгляда.

Двумя годами позже Джо получил свою первую работу в качестве разработчика программного обеспечения в Policy Management Systems Corporation, как программист начального уровня, разрабатывающий клиент-серверное приложение системы страхования для OS/2 и Presentation Manager. С годами он добавил к своему багажу знаний C++, Unix, Java, ASP, ASP.NET, C#, HTML, DHTML и XML, разрабатывая приложения для SCT, DocuCorp, IBM, Комитета по проведению олимпийских игр в Атланте, CheckFree, NCR, EDS, Delta Technology, Radiant Systems и Genuine Parts Company. Джо нравились творческие аспекты дизайна пользовательского интерфейса, и он осознал необходимость дисциплины при разработке программного обеспечения серверной стороны. Но когда у него была такая возможность, его любимым времяпрепровождением была отладка кода.

Сегодня Джо можно найти в Genuine Parts Company — родительской компании NAPA — в подразделении Automotive Parts Group Information Systems, где он трудится над своим детищем — веб-сайтом Storefront. Этот сайт обслуживает хранилища NAPA, предоставляя их счета и данные в сети систем AS/400. Связаться к Джо можно через его веб-сайт по адресу www.linqdev.com.

О техническом рецензенте

Фабио Клаудио Феррачати — старший консультант и аналитик-разработчик, имеющий дело с технологиями Microsoft. Он работает в компании Brain Forge (www.brainforce.com), в ее итальянском подразделении (www.brainforce.it). Является сертифицированным разработчиком решений Microsoft для .NET, сертифицированным разработчиком приложений Microsoft для .NET, сертифицированным профессионалом Microsoft, а также плодовитым автором и техническим рецензентом. В течение последних десяти лет он написал множество статей для итальянских и международных изданий, а также является соавтором более десятка книг на различные компьютерные темы.

Благодарности

Мы хотели бы поблагодарить Джона Скита (Jon Skeet), Джадсона Уайта (Judson White) и всех остальных сотрудников издательства Apress за совместную работу над этой книгой. В частности, благодарим Мери Тобин (Mary Tobin) за отслеживание работ и Эвана Бакингема (Ewan Buckingham) за реракитрование книги. Мы также выражаем благодарность Ким Уимпсетт (Kim Wimpsett) и Фабио Феррачати (Fabio Ferracchiati) за их усилия, которые сделали эту книгу намного лучше, чем она была в начале.

Адам Фримен и Джозеф Ратти-мл.

От издательства

Вы, читатель этой книги, и есть главный ее критик и комментатор. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать лучше и что еще вы хотели бы увидеть изданным нами. Нам интересно услышать и любые другие замечания, которые вам хотелось бы высказать в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо, либо просто посетить наш веб-сервер и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится или нет вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Посылая письмо или сообщение, не забудьте указать название книги и ее авторов, а также ваш обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию последующих книг.

Наши координаты:

E-mail: info@williamspublishing.com

WWW: <http://www.williamspublishing.com>

Информация для писем из:

России: 127055, г. Москва, ул. Лесная, д. 43, стр. 1

Украины: 03150, Киев, а/я 152

Исходный код примеров

Исходный код примеров, рассмотренных в книге, доступен для загрузки на сайте издательства по адресу <http://www.williamspublishing.com>.

ЧАСТЬ I

LINQ: ЯЗЫК ИНТЕГРИРОВАННЫХ ЗАПРОСОВ В C# 2010

В этой части

Глава 1. Знакомство с LINQ

Глава 2. Расширения языка C# для LINQ

ГЛАВА 1

Знакомство с LINQ

Листинг 1.1. Программа “Hello LINQ”

```
using System;
using System.Linq;
string[] greetings = {"hello world", "hello LINQ", "hello Apress"};
var items =
    from s in greetings
    where s.EndsWith("LINQ")
    select s;
foreach (var item in items)
    Console.WriteLine(item);
```

На заметку! Код из листинга 1.1 был добавлен в проект, созданный с помощью шаблона консольного приложения в Visual Studio 2010. Если это еще не сделано, добавьте директиву `using` для пространства имен `System.Linq`.

Запуск приведенного выше кода по нажатию <Ctrl+F5> выдаст следующий вывод в окно консоли:

```
hello LINQ
```

Сдвиг парадигмы

Вы почувствовали только что, как *ваш* мир сдвинулся с места? Как разработчик .NET, вы должны были это почувствовать. То, что продемонстрировано в тривиальном примере программы из листинга 1.1, похоже на запрос на языке структурированных запросов (Structured Query Language — SQL) к массиву строк¹. Взгляните на конструкцию `where`. Она выглядит так, будто использовался метод `EndsWith` объекта `string`, потому что так оно и есть. Может возникнуть вопрос: а как насчет типа переменной `var`? Выполняет ли по-прежнему компилятор C# контроль типов? Ответ — да, он проверяет статически типы во время компиляции. Какое средство или средства C# позволяют все это? Ответ: Microsoft Language Integrated Query (язык интегрированных запросов Microsoft), иначе называемый *LINQ*.

¹ Важно отметить, что порядок следования конструкций в запросе противоположен типичному SQL. К тому же добавлена часть `s in` запроса, которая представляет ссылку на набор элементов, содержащихся в источнике, которым в данном случае является массив строк "hello world", "hello LINQ" и "hello Apress".

Запрос к XML

В то время как пример из листинга 1.1 достаточно тривиален, пример в листинге 1.2 начинает отражать потенциальную мощь, которую вручает LINQ в руки разработчика .NET. Он демонстрирует легкость, с которой можно взаимодействовать и опрашивать данные XML (Extensible Markup Language — расширяемый язык разметки) с помощью API-интерфейса LINQ to XML. Обратите внимание, как на основе данных XML конструируется объект по имени `books`, с которым впоследствии можно взаимодействовать программно.

Листинг 1.2. Простой запрос к XML-разметке с использованием LINQ to XML

```
using System;
using System.Linq;
using System.Xml.Linq;
XElement books = XElement.Parse(
    @"<books>
      <book>
        <title>Pro LINQ: Language Integrated Query in C# 2010</title>
        <author>Joe Rattz</author>
      </book>
      <book>
        <title>Pro .NET 4.0 Parallel Programming in C#</title>
        <author>Adam Freeman</author>
      </book>
      <book>
        <title>Pro VB 2010 and the .NET 4.0 Platform</title>
        <author>Andrew Troelsen</author>
      </book>
    </books>");
var titles =
    from book in books.Elements("book")
    where (string) book.Element("author") == "Joe Rattz"
    select book.Element("title");
foreach(var title in titles)
    Console.WriteLine(title.Value);
```

На заметку! Код в листинге 1.2 требует добавления к ссылкам проекта сборки `System.Xml.Linq.dll`, если это еще не сделано. Также обратите внимание, что добавлена директива `using` для пространства имен `System.Xml.Linq`.

Запуск предыдущего кода нажатием `<Ctrl+F5>` приводит к выводу следующих данных в окно консоли:

```
Pro LINQ: Language Integrated Query in C# 2010
```

Обратите внимание на то, как данные XML были разобраны для помещения в объект `XElement`. Объект `XmlDocument` нигде не создавался. Среди преимуществ LINQ to XML — расширения, которые он привносит в XML API. Теперь вместо того, чтобы сосредоточивать все вокруг `XmlDocument`, как того требует W3C Document Object Model (DOM) XML API, интерфейс LINQ to XML позволяет разработчику взаимодействовать на уровне элемента, используя класс `XElement`.

На заметку! В дополнение к средствам запросов, LINQ to XML предоставляет более мощный и простой способ использования интерфейса для работы с данными XML.

Здесь также применялся SQL-подобный синтаксис для опроса данных XML, как если бы это была база данных.

Запрос к базе данных SQL Server

В следующем примере демонстрирует использование LINQ to SQL для опроса таблиц базы данных. В коде из листинга 1.3 выполняется запрос к стандартной базе данных примеров Microsoft Northwind.

Листинг 1.3. Простой запрос XML с использованием LINQ to XML

```
using System.Linq;
using System.Data.Linq;
using nwind;
Northwind db = new Northwind(@"Data Source=.\SQLEXPRESS;Initial Catalog=Northwind");
var custs =
    from c in db.Customers
    where c.City == "Rio de Janeiro"
    select c;
foreach (var cust in custs)
    Console.WriteLine("{0}", cust.CompanyName);
```

На заметку! Код в листинге 1.3 требует добавления сборки `System.Data.Linq.dll` к списку ссылок проекта, если это не сделано ранее. Также обратите внимание, что добавлена директива `using` для пространства имен `System.Xml.Linq`.

В коде была добавлена директива `using` для пространства имен `nwind`. В этом примере должна использоваться утилита командной строки `SQLMetal` или `Object Relational Designer`, чтобы сгенерировать сущностные классы для целевой базы данных, которой в данном случае является база данных примеров Microsoft Northwind. В главе 12 показано, как это делать с `SQLMetal`. Сгенерированные сущностные классы создаются в пространстве имен `nwind`, которое указано при их генерации. После этого в проект добавляется сгенерированный `SQLMetal` исходный модуль, а также директива `using` для пространства имен `nwind`.

На заметку! Для корректной установки соединения может понадобиться изменить строку соединения, переданную конструктору `Northwind` в листинге 1.3. В разделе "`DataContext()` и `[Your] DataContext()`" главы 16 описаны разные способы подключения к базе данных.

Запуск предыдущего кода нажатием `<Ctrl+F5>` приводит к выводу следующих данных в окно консоли:

```
Hanari Carnes
Que Delicia
Ricardo Adocicados
```

В этом примере демонстрируется опрос таблицы `Customers` из базы данных Northwind на предмет списка заказчиков из Рио-де-Жанейро (Rio de Janeiro). Хотя может показаться, что здесь не происходит ничего нового или особенного, чего нельзя было бы получить существующими средствами, все же есть серьезные отличия. Важнее всего то, что этот запрос интегрирован в язык, а это значит, что получается поддержка уровня языка, включающая проверку синтаксиса и средство `IntelliSense`. Ушли в прошлое те дни, когда запрос `SQL` записывался в строку, и ошибку невозможно было обнаружить вплоть до выполнения кода. Вы хотите сделать конструкцию `where` зависящей от поля