

Содержание

Предисловие	15
Об авторе	19
Введение	20
Часть I. Отправляемся в путь	25
Глава 1. Изменение способа разработки на C#	27
1.1. Начнем с простого типа данных	28
1.1.1. Тип Product в C# 1	28
1.1.2. Строго типизированные коллекции в C# 2	29
1.1.3. Автоматически реализуемые свойства в C# 3	30
1.1.4. Именованные аргументы в C# 4	31
1.2. Сортировка и фильтрация	33
1.2.1. Сортировка продуктов по названию	33
1.2.2. Запрос коллекций	35
1.3. Обработка отсутствия данных	37
1.3.1. Представление неизвестной цены	37
1.3.2. Необязательные параметры и значения по умолчанию	38
1.4. Введение в LINQ	39
1.4.1. Выражения запросов и внутренние запросы	39
1.4.2. Запросы XML	40
1.4.3. Язык LINQ to SQL	41
1.5. Модель COM и динамическая типизация	42
1.5.1. Упрощение совместимости с COM	42
1.5.2. Взаимодействие с динамическим языком	43
1.6. Разделение платформы .NET	44
1.6.1. Язык C#	44
1.6.2. Среда исполнения	45
1.6.3. Библиотеки инфраструктуры	45
1.7. Как сделать ваш код потрясающим	46
1.7.1. Представление целых программ как фрагментов	46
1.7.2. Дидактический код — это не рабочий код	47
1.7.3. Ваш новый лучший друг: спецификация языка	48
1.8. Резюме	48
Глава 2. Язык C# 1 — основа основ	49
2.1. Делегаты	50
2.1.1. Рецепт для простых делегатов	50
2.1.2. Объединение и удаление делегатов	55
2.1.3. Кратко о событиях	56
2.1.4. Резюме по делегатам	57
2.2. Характеристики системы типов	57
2.2.1. Место языка C# в мире систем типов	58
2.2.2. Когда система типов C# 1 недостаточно богата	61

2.2.3. Резюме по характеристикам системы типов	63
2.3. Типы значений и ссылочные типы	64
2.3.1. Значения и ссылки в реальном мире	64
2.3.2. Основные принципы ссылочных типов и типов значений	65
2.3.3. Разрушение мифов	66
2.3.4. Упаковка и распаковка	68
2.3.5. Резюме по типам значений и ссылочным типам	69
2.4. Вне C# 1: новые возможности на солидной базе	70
2.4.1. Средства, связанные с делегатами	70
2.4.2. Средства, связанные с системой типов	72
2.4.3. Средства, связанные с типами значений	74
2.5. Резюме	75
Часть II. Язык C# 2: решение проблем языка C# 1	77
Глава 3. Параметрическая типизация с обобщениями	79
3.1. Почему необходимы обобщения	80
3.2. Простые обобщения для повседневного использования	81
3.2.1. Обучение на примере: обобщенный словарь	82
3.2.2. Обобщенные типы и параметры типа	83
3.2.3. Обобщенные методы и чтение объявлений обобщений	86
3.3. Дополнительные сведения	90
3.3.1. Ограничения типов	90
3.3.2. Выведение типов для аргументов типа обобщенных методов	95
3.3.3. Реализация обобщений	96
3.4. Расширенные обобщения	102
3.4.1. Статические поля и статические конструкторы	102
3.4.2. Как компилятор JIT обрабатывает обобщения	104
3.4.3. Перебор обобщений	106
3.4.4. Рефлексия и обобщения	108
3.5. Ограничения, налагаемые на обобщения в C# и других языках	112
3.5.1. Недостаток обобщенной вариантности	112
3.5.2. Недостаток ограничений оператора или “числового” ограничения	117
3.5.3. Недостаток обобщенных свойств, индексаторов и других элементов типа	119
3.5.4. Сравнение с шаблонами C++	119
3.5.5. Сравнение с обобщениями в языке Java	120
3.6. Резюме	122
Глава 4. Типы, допускающие значения null	123
4.1. Что вы делаете, когда у вас просто нет значения	123
4.1.1. Почему переменные типа значений не могут содержать значение null	124
4.1.2. Шаблоны для представления значений null в языке C# 1	125
4.2. Типы System.Nullable<T> и System.Nullable	127
4.2.1. Введение в тип Nullable<T>	127
4.2.2. Упаковка и распаковка типа Nullable<T>	130
4.2.3. Равенство экземпляров Nullable<T>	131
4.2.4. Поддержка необобщенного класса Nullable	132
4.3. Синтаксис языка C# 2 для типов, допускающих значения null	133

8	Содержание	
4.3.1.	Модификатор ?	133
4.3.2.	Присвоение и сравнение со значением null	134
4.3.3.	Преобразования и операторы типов, допускающих значения null	136
4.3.4.	Логика типов, допускающих значения null	139
4.3.5.	Использование оператора as с типами, допускающими значения null	141
4.3.6.	Объединяющий null-оператор	141
4.4.	Новые способы использования типов, допускающих значения null	144
4.4.1.	Попытка работы без использования выходных параметров	144
4.4.2.	Облегченные сравнения с использованием объединяющего null-оператора	146
4.5.	Резюме	149
	Глава 5. Скоростные делегаты	151
5.1.	Попрощайтесь с неуклюжим синтаксисом делегата	152
5.2.	Преобразования группы методов	153
5.3.	Ковариация и контрвариация	155
5.3.1.	Контрвариация для параметров делегата	155
5.3.2.	Ковариация типов возвращаемого значения делегата	157
5.3.3.	Небольшой риск несовместимости	158
5.4.	Встраиваемые действия делегата с анонимными методами	159
5.4.1.	Начнем с простого: действие с параметром	159
5.4.2.	Возвращение значений из анонимных методов	161
5.4.3.	Игнорирование параметров делегата	163
5.5.	Захват переменных в анонимных методах	164
5.5.1.	Определение замкнутых выражений и переменных различных типов	165
5.5.2.	Исследование поведения захваченных переменных	166
5.5.3.	В чем смысл захваченных переменных	167
5.5.4.	Продление существования захваченных переменных	168
5.5.5.	Создание экземпляра локальной переменной	169
5.5.6.	Комбинация совместно используемых и независимых переменных	171
5.5.7.	Правила захвата переменных и резюме	173
5.6.	Резюме	174
	Глава 6. Простой путь реализации итераторов	175
6.1.	C# 1: сложность самодельных итераторов	176
6.2.	C# 2: простые итераторы с операторами yield	179
6.2.1.	Блоки итератора и yield return	179
6.2.2.	Визуализация рабочего потока итератора	181
6.2.3.	Расширенный поток выполнения итератора	183
6.2.4.	Причуды реализации	186
6.3.	Реальные примеры итератора	187
6.3.1.	Перебор дат в расписании	187
6.3.2.	Перебор строк в файле	188
6.3.3.	Фильтрация элементов при “ленивом” использовании блока итератора и предиката	191
6.4.	Псевдосинхронный код и библиотека параллельных вычислений и координации	193
6.5.	Резюме	195

Глава 7. Заключение С# 2: финальные средства	197
7.1. Разделяемые типы	198
7.1.1. Создание типа в нескольких файлах	199
7.1.2. Использование разделяемых типов	201
7.1.3. Разделяемые методы — только С# 3!	202
7.2. Статические классы	204
7.3. Разные степени доступа методов получения и установки значения свойства	206
7.4. Псевдонимы пространства имен	207
7.4.1. Квалификация псевдонимов пространства имен	208
7.4.2. Глобальный псевдоним пространства имен	209
7.4.3. Внешние псевдонимы	210
7.5. Директивы pragma	211
7.5.1. Директива pragma warning	211
7.5.2. Директива pragma checksum	212
7.6. Буферы фиксированного размера в небезопасном коде	213
7.7. Демонстрация внутренних членов избранным сборкам	215
7.7.1. Дружественные сборки в простом случае	215
7.7.2. Зачем использовать атрибут InternalsVisibleTo	216
7.7.3. Атрибут InternalsVisibleTo и подписанные сборки	217
7.8. Резюме	217
Часть III. Язык С# 3: революция в программировании	219
Глава 8. Интеллектуальный компилятор	221
8.1. Автоматически реализованные свойства	222
8.2. Неявная типизация локальных переменных	224
8.2.1. Использование ключевого слова var для объявления локальной переменной	224
8.2.2. Ограничения неявной типизации	226
8.2.3. Преимущества и недостатки неявной типизации	227
8.2.4. Рекомендации	228
8.3. Упрощенная инициализация	229
8.3.1. Определение демонстрационных типов	229
8.3.2. Установка простых свойств	230
8.3.3. Установка свойств внедренных объектов	231
8.3.4. Инициализаторы коллекции	232
8.3.5. Использование средств инициализации	235
8.4. Неявно типизированные массивы	236
8.5. Анонимные типы	237
8.5.1. Первая встреча с анонимными типами	237
8.5.2. Члены анонимных типов	239
8.5.3. Инициализатор проекции	240
8.5.4. Какой в этом смысл?	241
8.6. Резюме	242
Глава 9. Лямбда-выражения и деревья выражений	243
9.1. Лямбда-выражения как делегаты	244
9.1.1. Предварительный выбор: представление типа делегата Func<...>	245

10	Содержание	
9.1.2.	Первое преобразование в лямбда-выражение	245
9.1.3.	Использование одиночного выражения как тела	247
9.1.4.	Неявно типизированные списки параметров	247
9.1.5.	Сокращение для одиночного параметра	247
9.2.	Простой пример использования типа List<T> и событий	249
9.2.1.	Фильтрация, сортировка и действия со списками	249
9.2.2.	Регистрация в обработчике событий	251
9.3.	Деревья выражений	252
9.3.1.	Программное построение деревьев выражений	252
9.3.2.	Компиляция деревьев выражений в делегаты	253
9.3.3.	Преобразование лямбда-выражений C# в деревья выражений	255
9.3.4.	Деревья выражений — основа LINQ	258
9.3.5.	Деревья выражений вне LINQ	259
9.4.	Изменения в механизме вывода типов и разрешения перегрузки	261
9.4.1.	Причины изменений: упрощение вызова обобщенных методов	261
9.4.2.	Выведение типа возвращаемого значения анонимных функций	262
9.4.3.	Двухэтапное выведение типов	264
9.4.4.	Выбор правильной версии перегруженного метода	267
9.4.5.	Оболочка вывода типов и разрешения перегрузки	269
9.5.	Резюме	269
	Глава 10. Методы расширения	271
10.1.	Жизнь до методов расширения	272
10.2.	Синтаксис метода расширения	274
10.2.1.	Объявление методов расширения	274
10.2.2.	Вызов методов расширения	275
10.2.3.	Поиск метода расширения	277
10.2.4.	Вызов метода по пустой ссылке	278
10.3.	Методы расширения в .NET 3.5	279
10.3.1.	Первые шаги с классом Enumerable	280
10.3.2.	Фильтрация при помощи метода Where и сцепления вызовов методов	281
10.3.3.	Интерлюдия: разве мы не видели метод Where прежде?	283
10.3.4.	Проецирование с использованием метода Select и анонимных типов	284
10.3.5.	Сортировка с использованием метода OrderBy	285
10.3.6.	Бизнес-примеры с использованием сцепления	286
10.4.	Идеи и правила применения	287
10.4.1.	“Расширение мира” и обогащение интерфейсов	288
10.4.2.	Текущие интерфейсы	288
10.4.3.	Разумное использование методов расширения	290
10.5.	Резюме	291
	Глава 11. Выражения запросов и LINQ to Objects	293
11.1.	Введение в LINQ	294
11.1.1.	Фундаментальные концепции LINQ	294
11.1.2.	Определение эталонной модели данных	299
11.2.	Сначала простое: выбор элементов	300
11.2.1.	Начнем с источника и закончим выбором	300
11.2.2.	Трансляция компилятора как основа выражений запроса	301
11.2.3.	Переменные диапазона и нетривиальные проекции	304

11.2.4. Операторы Cast, OfType и явно типизированные переменные диапазона	305
11.3. Фильтрация и упорядочение последовательности	307
11.3.1. Фильтрация с использованием директивы where	308
11.3.2. Вырожденное выражение запроса	308
11.3.3. Упорядочение с использованием директивы orderby	309
11.4. Директива let и прозрачные идентификаторы	311
11.4.1. Промежуточное вычисление с директивой let	311
11.4.2. Прозрачные идентификаторы	312
11.5. Объединения	314
11.5.1. Внутренние объединения с использованием директивы join	314
11.5.2. Групповое объединение и директива join...into	318
11.5.3. Перекрестное объединение и сглаживание последовательности с использованием нескольких директив from	321
11.6. Группировки и продолжения	324
11.6.1. Группировка при помощи директивы group...by	324
11.6.2. Продолжения запроса	327
11.7. Выбор между выражениями запроса и точечной формой записи	330
11.7.1. Операции, требующие точечной формы записи	330
11.7.2. Выражения запроса, где точечная форма записи может оказаться проще	331
11.7.3. В каких случаях выбирать выражения запроса	331
11.8. Резюме	332
Глава 12. LINQ вне коллекций	335
12.1. Запрос к базе данных при помощи LINQ to SQL	336
12.1.1. Первые шаги: база данных и модель	336
12.1.2. Исходные запросы	338
12.1.3. Запросы, задействующие объединения	341
12.2. Преобразования с использованием интерфейсов IQueryable и IQueryableProvider	343
12.2.1. Знакомство с интерфейсом IQueryable<T> и связанными интерфейсами	343
12.2.2. Имитация: реализация интерфейса для регистрации вызовов	345
12.2.3. Склеивание выражений: методы расширения класса Queryable	347
12.2.4. Имитационный провайдер запроса в действии	349
12.2.5. Оболочка IQueryable	350
12.3. Дружественные API LINQ и LINQ to XML	351
12.3.1. Базовые типы в LINQ to XML	351
12.3.2. Декларативное создание	353
12.3.3. Запросы одиночных узлов	355
12.3.4. Сглаженные операторы запроса	357
12.3.5. Работа в гармонии с LINQ	358
12.4. Замена LINQ to Objects на Parallel LINQ	359
12.4.1. Рисование множества Мандельброта в одном потоке	359
12.4.2. Знакомство с ParallelEnumerable, ParallelQuery и AsParallel	361
12.4.3. Настройка параллельных запросов	362
12.5. Инверсия модели запросов при помощи LINQ to Rx	364
12.5.1. Интерфейсы IObservable<T> и IObservable<T>	364
12.5.2. Начнем с простого (снова)	366
12.5.3. Запрос Observables	367
12.5.4. В чем смысл	369

12 Содержание

12.6. Расширение LINQ to Objects	370
12.6.1. Проектирование и правила реализации	370
12.6.2. Пример расширения: выбор произвольного элемента	372
12.7. Резюме	373

Часть IV. C# 4: приятно поиграть с другими 375

Глава 13. Небольшие изменения для упрощения кода 377

13.1. Необязательные параметры и именованные аргументы	378
13.1.1. Необязательные параметры	378
13.1.2. Именованные аргументы	384
13.1.3. Объединение средств	387
13.2. Усовершенствования для совместимости COM	392
13.2.1. Ужасы автоматизации Word до C# 4	392
13.2.2. Месть необязательных параметров и именованных аргументов	393
13.2.3. Когда параметр ref не является ссылочным параметром	394
13.2.4. Вызов именованных индексов	394
13.2.5. Связывание основных сборок взаимодействия	396
13.3. Обобщенная вариантность для интерфейсов и делегатов	399
13.3.1. Типы вариантности: ковариация и контрвариация	399
13.3.2. Использование вариантности в интерфейсах	400
13.3.3. Использование вариантности в делегатах	403
13.3.4. Сложные ситуации	404
13.3.5. Ограничения и замечания	405
13.4. Очень маленькие изменения в блокировке и полеподобных событиях	409
13.4.1. Надежная блокировка	409
13.4.2. Изменения в полеподобных событиях	410
13.5. Резюме	410

Глава 14. Динамическое связывание в статическом языке 413

14.1. Что? Когда? Почему? Как?	414
14.1.1. Что такое динамическая типизация	415
14.1.2. Когда полезна динамическая типизация и почему	415
14.1.3. Как C# 4 поддерживает динамическую типизацию	417
14.2. Пятиминутное руководство по динамике	417
14.3. Примеры динамической типизации	420
14.3.1. Модель COM вообще и Microsoft Office в частности	420
14.3.2. Динамические языки, такие как IronPython	422
14.3.3. Динамическая типизация в простом управляемом коде	426
14.4. Заглянем за кулисы	432
14.4.1. Знакомство с исполняющей средой динамического языка	432
14.4.2. Базовые концепции DLR	433
14.4.3. Как компилятор C# справляется с динамикой	436
14.4.4. Компилятор C# становится еще умнее	440
14.4.5. Ограничения на динамический код	443
14.5. Реализация динамического поведения	446
14.5.1. Использование класса ExpandableObject	446
14.5.2. Использование класса DynamicObject	450

14.5.3. Реализация интерфейса IDynamicMetaObjectProvider	456
14.6. Резюме	460
Глава 15. Позволим нашему коду выразаться яснее с помощью Code Contracts	463
15.1. Жизнь до Code Contracts	464
15.2. Знакомство с Code Contracts	466
15.2.1. Предусловия	468
15.2.2. Постусловия	468
15.2.3. Инварианты	470
15.2.4. Утверждения и предположения	471
15.2.5. Устаревшие контракты	473
15.3. Перезапись бинарного кода при помощи <code>ccrewrite</code> и <code>ccrefgen</code>	474
15.3.1. Простая перезапись	475
15.3.2. Наследование контракта	476
15.3.3. Сборки ссылок на контракт	479
15.3.4. Поведение при нарушении	480
15.4. Статическая проверка	482
15.4.1. Знакомство со статической проверкой	483
15.4.2. Неявные обязательства	485
15.4.3. Выборочная проверка	488
15.5. Документирование контрактов при помощи <code>ccdocgen</code>	490
15.6. Практические контракты	492
15.6.1. Философия: что в контракте?	493
15.6.2. С чего начинать	494
15.6.3. Возможности, всюду возможности	495
15.7. Резюме	498
Глава 16. Что дальше	501
16.1. C# — комбинация традиций и новшеств	501
16.2. Информатика и .NET	502
16.3. Мир компьютеров	503
16.4. Счастливого пути!	504
Приложение А. Стандартные операторы запроса LINQ	505
А.1. Объединение	505
А.2. Конкатенация	506
А.3. Преобразование	506
А.4. Операторы элементов	508
А.5. Равенство	509
А.6. Генерация	510
А.7. Группировка	511
А.8. Объединения	511
А.9. Разделение	512
А.10. Проекция	513
А.11. Кванторы	514
А.12. Фильтрация	514
А.13. Операторы на базе набора	515
А.14. Сортировка	516

14 Содержание

Приложение Б. Обобщенные коллекции в .NET	517
Б.1. Интерфейсы	517
Б.2. Списки	519
Б.2.1. Тип List<T>	519
Б.2.2. Массивы	520
Б.2.3. Тип LinkedList<T>	521
Б.2.4. Типы Collection<T>, BindingList<T>, ObservableCollection<T> и KeyedCollection<TKey,TItem>	522
Б.2.5. Типы ReadOnlyCollection<T> и ReadOnlyObservableCollection<T>	523
Б.3. Словари	523
Б.3.1. Тип Dictionary<TKey,TValue>	523
Б.3.2. Типы SortedList<TKey,TValue> и SortedDictionary<TKey,TValue>	524
Б.4. Наборы	525
Б.4.1. Тип HashSet<T>	525
Б.4.2. Тип SortedSet<T> (.NET 4)	525
Б.5. Типы Queue<T> и Stack<T>	526
Б.5.1. Тип Queue<T>	526
Б.5.2. Тип Stack<T>	527
Б.6. Параллельные коллекции (.NET 4)	527
Б.6.1. Типы IProducerConsumerCollection<T> и BlockingCollection<T>	527
Б.6.2. Типы ConcurrentBag<T>, ConcurrentQueue<T>, ConcurrentStack<T>	528
Б.6.3. Тип ConcurrentDictionary<TKey,TValue>	528
Б.7. Резюме	528
Приложение В. Итог по версиям	531
В.1. Главные выпуски инфраструктуры рабочего стола	531
В.2. Средства языка C#	532
В.2.1. C# 2.0	532
В.2.2. C# 3.0	532
В.2.3. C# 4.0	533
В.3. Средства библиотеки	533
В.3.1. .NET 2.0	533
В.3.2. .NET 3.0	533
В.3.3. .NET 3.5	534
В.3.4. .NET 4	534
В.4. Средства исполняющей среды (CLR)	535
В.4.1. CLR 2.0	535
В.4.2. CLR 4.0	536
В.5. Сопутствующие инфраструктуры	536
В.5.1. Compact Framework	536
В.5.2. Silverlight	536
В.5.3. Micro Framework	537
В.6. Резюме	537
Предметный указатель	539