

Содержание

Об авторе	32
О техническом редакторе	32
Благодарности	32
Введение	33
Автор и читатели — одна команда	33
Краткий обзор книги	34
Часть I. Введение в C# и платформу .NET	34
Часть II. Основы программирования на C#	34
Часть III. Объектно-ориентированное программирование на C#	35
Часть IV. Дополнительные конструкции программирования на C#	36
Часть V. Программирование с использованием сборок .NET	37
Часть VI. Введение в библиотеки базовых классов .NET	38
Часть VII. Windows Presentation Foundation	40
Часть VIII. ASP.NET Web Forms	41
Загружаемые приложения	41
Исходный код примеров	42
От издательства	42
Часть I. Введение в C# и платформу .NET	43
Глава 1. Философия .NET	44
Начальное знакомство с платформой .NET	44
Некоторые основные преимущества платформы .NET	45
Введение в строительные блоки платформы .NET (CLR, CTS и CLS)	46
Роль библиотек базовых классов	46
Что привносит язык C#	47
Сравнение управляемого и неуправляемого кода	49
Другие языки программирования, ориентированные на .NET	49
Жизнь в многоязычном окружении	50
Обзор сборок .NET	51
Роль языка CIL	52
Роль метаданных типов .NET	55
Роль манифеста сборки	56
Понятие общей системы типов (CTS)	56
Типы классов CTS	57
Типы интерфейсов CTS	57
Типы структур CTS	58
Типы перечислений CTS	58
Типы делегатов CTS	59
Члены типов CTS	59
Встроенные типы данных CTS	59
Понятие общезыковой спецификации (CLS)	60
Обеспечение совместимости с CLS	62
Понятие общезыковой исполняющей среды (CLR)	62
Различия между сборками, пространствами имен и типами	64
Роль корневого пространства имен Microsoft	67
Доступ к пространству имен программным образом	67
Ссылка на внешние сборки	68

Исследование сборки с помощью утилиты <code>ildasm.exe</code>	69
Просмотр CIL-кода	70
Просмотр метаданных типов	70
Просмотр метаданных сборки (манифеста)	71
Независимая от платформы природа .NET	71
Несколько слов по поводу приложений Windows 8	73
Построение приложений в стиле Metro	74
Роль .NET в среде Windows 8	75
Резюме	77
Глава 2. Создание приложений на языке C#	78
Роль .NET Framework 4.5 SDK	78
Окно командной строки разработчика	79
Построение приложений C# с использованием <code>csc.exe</code>	79
Указание целевых входных и выходных параметров	80
Ссылка на внешние сборки	82
Ссылка на несколько внешних сборок	83
Компиляция нескольких файлов исходного кода	83
Работа с ответными файлами в C#	84
Построение приложений .NET с использованием Notepad++	85
Построение приложений .NET с помощью SharpDevelop	86
Создание простого тестового проекта	87
Построение приложений .NET с помощью Visual C# Express	89
Некоторые уникальные возможности Visual C# Express	89
Построение приложений .NET с помощью Visual Studio	89
Некоторые уникальные возможности Visual Studio	90
Выбор целевой версии .NET Framework в диалоговом окне New Project	91
Использование утилиты Solution Explorer	91
Утилита Class View	93
Утилита Object Browser	93
Встроенная поддержка рефакторинга кода	94
Фрагменты кода и технология Surround With	96
Утилита Class Designer	98
Интегрированная система документации .NET Framework 4.5 SDK	100
Резюме	102
Часть II. Основы программирования на C#	103
Глава 3. Главные конструкции программирования на C#: часть I	104
Структура простой программы C#	104
Вариации метода <code>Main()</code>	106
Указание кода ошибки приложения	107
Обработка аргументов командной строки	108
Указание аргументов командной строки в Visual Studio	109
Интересное отступление от темы: некоторые дополнительные члены класса <code>System.Environment</code>	110
Класс <code>System.Console</code>	111
Базовый ввод-вывод с помощью класса <code>Console</code>	112
Форматирование консольного вывода	113
Форматирование числовых данных	114
Форматирование числовых данных в приложениях, отличных от консольных	115
Системные типы данных и соответствующие ключевые слова C#	115
Объявление и инициализация переменных	117

8 Содержание

Внутренние типы данных и операция <code>new</code>	118
Иерархия классов для типов данных	119
Члены числовых типов данных	120
Члены <code>System.Boolean</code>	121
Члены <code>System.Char</code>	121
Синтаксический разбор значений из строковых данных	121
Типы <code>System.DateTime</code> и <code>System.TimeSpan</code>	122
Сборка <code>System.Numerics.dll</code>	122
Работа со строковыми данными	124
Базовые манипуляции строками	125
Конкатенация строк	125
Управляющие последовательности	126
Определение дословных строк	127
Строки и равенство	127
Строки являются неизменяемыми	128
Тип <code>System.Text.StringBuilder</code>	129
Сужающие и расширяющие преобразования типов данных	130
Ключевое слово <code>checked</code>	133
Настройка проверки переполнения на уровне всего проекта	134
Ключевое слово <code>unchecked</code>	135
Понятие неявно типизированных локальных переменных	136
Ограничения неявно типизированных переменных	137
Неявно типизированные данные являются строго типизированными	138
Польза от неявно типизированных локальных переменных	138
Итерационные конструкции C#	139
Цикл <code>for</code>	140
Цикл <code>foreach</code>	140
Циклы <code>while</code> и <code>do/while</code>	141
Конструкции принятия решений и операции равенства/сравнения	142
Оператор <code>if/else</code>	142
Операции равенства и сравнения	142
Условные операции	143
Оператор <code>switch</code>	143
Резюме	145
Глава 4. Главные конструкции программирования на C#: часть II	146
Методы и модификаторы параметров	146
Стандартное поведение передачи параметров по значению	147
Модификатор <code>out</code>	148
Модификатор <code>ref</code>	149
Модификатор <code>params</code>	150
Определение необязательных параметров	151
Вызов методов с использованием именованных параметров	152
Понятие перегрузки методов	154
Массивы в C#	156
Синтаксис инициализации массивов C#	157
Неявно типизированные локальные массивы	158
Определение массива объектов	158
Работа с многомерными массивами	159
Использование массивов в качестве аргументов и возвращаемых значений	160
Базовый класс <code>System.Array</code>	161
Тип <code>enum</code>	162
Управление хранилищем, лежащим в основе перечисления	163

Объявление переменных типа перечисления	164
Тип <code>System.Enum</code>	165
Динамическое извлечение пар “имя/значение” перечисления	166
Типы структур	168
Создание переменных типа структур	169
Типы значений и ссылочные типы	170
Типы значений, ссылочные типы и операция присваивания	171
Типы значений, содержащие ссылочные типы	173
Передача ссылочных типов по значению	174
Передача ссылочных типов по ссылке	175
Заключительные детали относительно типов значений и ссылочных типов	176
Понятие типов, допускающих <code>null</code> , в C#	177
Работа с типами, допускающими <code>null</code>	178
Операция ??	179
Резюме	180
Часть III. Объектно-ориентированное программирование на C#	181
Глава 5. Инкапсуляция	182
Знакомство с типом класса C#	182
Размещение объектов с помощью ключевого слова <code>new</code>	184
Понятие конструкторов	185
Роль стандартного конструктора	185
Определение специальных конструкторов	186
Еще раз о стандартном конструкторе	187
Роль ключевого слова <code>this</code>	188
Построение цепочки вызовов конструкторов с использованием <code>this</code>	189
Обзор потока конструктора	192
Еще раз о необязательных аргументах	193
Понятие ключевого слова <code>static</code>	194
Определение статических полей данных	195
Определение статических методов	197
Определение статических конструкторов	198
Определение статических классов	200
Основные принципы объектно-ориентированного программирования	201
Роль инкапсуляции	201
Роль наследования	202
Роль полиморфизма	203
Модификаторы доступа C#	204
Стандартные модификаторы доступа	205
Модификаторы доступа и вложенные типы	206
Первый принцип: службы инкапсуляции C#	206
Инкапсуляция с использованием традиционных методов доступа и изменения	207
Инкапсуляция с использованием свойств <code>.NET</code>	209
Использование свойств внутри определения класса	212
Свойства, доступные только для чтения и только для записи	213
Еще раз о ключевом слове <code>static</code> : определение статических свойств	214
Понятие автоматических свойств	215
Взаимодействие с автоматическими свойствами	216
Замечания относительно автоматических свойств и стандартных значений	217
Понятие синтаксиса инициализации объектов	218
Вызов специальных конструкторов с помощью синтаксиса инициализации	220
Инициализация вложенных типов	221

10 Содержание

Работа с данными константных полей	222
Понятие полей, допускающих только чтение	223
Статические поля, допускающие только чтение	224
Понятие частичных типов	224
Резюме	226
Глава 6. Понятие наследования и полиморфизма	227
Базовый механизм наследования	227
Указание родительского класса для существующего класса	228
Замечание относительно множества базовых классов	230
Ключевое слово <code>sealed</code>	230
Изменение диаграмм классов Visual Studio	231
Второй принцип ООП: подробности о наследовании	233
Управление созданием базового класса с помощью ключевого слова <code>base</code>	234
Хранение секретов семейства: ключевое слово <code>protected</code>	236
Добавление запечатанного класса	237
Реализация модели включения/делегации	237
Определения вложенных типов	239
Третий принцип ООП: поддержка полиморфизма в C#	240
Ключевые слова <code>virtual</code> и <code>override</code>	241
Переопределение виртуальных членов в IDE-среде Visual Studio	243
Запечатывание виртуальных членов	244
Абстрактные классы	244
Понятие полиморфного интерфейса	246
Сокрытие членов	249
Правила приведения к базовому и производному классу	251
Ключевое слово <code>as</code>	252
Ключевое слово <code>is</code>	253
Главный родительский класс <code>System.Object</code>	253
Переопределение <code>System.Object.ToString()</code>	256
Переопределение <code>System.Object.Equals()</code>	257
Переопределение <code>System.Object.GetHashCode()</code>	258
Тестирование модифицированного класса <code>Person</code>	259
Статические члены <code>System.Object</code>	260
Резюме	260
Глава 7. Структурированная обработка исключений	261
Ода ошибкам и исключениям	261
Роль обработки исключений .NET	262
Строительные блоки обработки исключений в .NET	263
Базовый класс <code>System.Exception</code>	264
Простейший пример	265
Генерация общего исключения	267
Перехват исключений	268
Конфигурирование состояния исключения	269
Свойство <code>TargetSite</code>	269
Свойство <code>StackTrace</code>	270
Свойство <code>HelpLink</code>	271
Свойство <code>Data</code>	271
Исключения уровня системы (<code>System.SystemException</code>)	273
Исключения уровня приложения (<code>System.ApplicationException</code>)	274
Построение специальных исключений, способ первый	274

Построение специальных исключений, способ второй	276
Построение специальных исключений, способ третий	277
Обработка нескольких исключений	278
Общие операторы catch	280
Повторная генерация исключений	281
Внутренние исключения	282
Блок finally	283
Какие исключения могут генерировать методы?	283
Результат наличия необработанных исключений	284
Отладка необработанных исключений с использованием Visual Studio	285
Резюме	285
Глава 8. Работа с интерфейсами	287
Понятие интерфейсных типов	287
Сравнение интерфейсных типов и абстрактных базовых классов	288
Определение специальных интерфейсов	290
Реализация интерфейса	292
Вызов членов интерфейса на уровне объектов	294
Получение ссылок на интерфейсы: ключевое слово as	295
Получение ссылок на интерфейсы: ключевое слово is	295
Использование интерфейсов в качестве параметров	296
Использование интерфейсов в качестве возвращаемых значений	298
Массивы интерфейсных типов	298
Реализация интерфейсов с использованием Visual Studio	299
Явная реализация интерфейсов	301
Проектирование иерархий интерфейсов	303
Множественное наследование посредством интерфейсных типов	304
Интерфейсы IEnumerable и IEnumerator	306
Построение методов итератора с применением ключевого слова yield	308
Построение именованного итератора	310
Интерфейс ICloneable	311
Более сложный пример клонирования	313
Интерфейс IComparable	315
Указание нескольких порядков сортировки посредством IComparer	318
Специальные свойства и специальные типы сортировки	319
Резюме	320
Часть IV. Дополнительные конструкции программирования на C#	321
Глава 9. Коллекции и обобщения	322
Побудительные причины создания классов коллекций	322
Пространство имен System.Collections	324
Обзор пространства имен System.Collections.Specialized	326
Проблемы, связанные с необобщенными коллекциями	327
Проблема производительности	327
Проблемы с безопасностью типов	330
Первый взгляд на обобщенные коллекции	333
Роль параметров обобщенных типов	334
Указание параметров типа для обобщенных классов и структур	335
Указание параметров типа для обобщенных членов	336
Указание параметров типов для обобщенных интерфейсов	336

12 Содержание

Пространство имен <code>System.Collections.Generic</code>	338
Синтаксис инициализации коллекций	339
Работа с классом <code>List<T></code>	340
Работа с классом <code>Stack<T></code>	342
Работа с классом <code>Queue<T></code>	342
Работа с классом <code>SortedSet<T></code>	344
Пространство имен <code>System.Collections.ObjectModel</code>	345
Работа с <code>ObservableCollection<T></code>	345
Создание специальных обобщенных методов	347
Выведение параметров типа	349
Создание специальных обобщенных структур и классов	351
Ключевое слово <code>default</code> в обобщенном коде	352
Ограничение параметров типа	353
Примеры использования ключевого слова <code>where</code>	353
Недостаток ограничений операций	355
Резюме	356
Глава 10. Делегаты, события и лямбда-выражения	357
Понятие типа делегата <code>.NET</code>	357
Определение типа делегата в <code>C#</code>	358
Базовые классы <code>System.MulticastDelegate</code> и <code>System.Delegate</code>	360
Пример простейшего делегата	362
Исследование объекта делегата	363
Отправка уведомлений о состоянии объекта с использованием делегатов	364
Включение группового вызова	367
Удаление целей из списка вызовов делегата	368
Синтаксис групповых преобразований методов	369
Понятие обобщенных делегатов	370
Обобщенные делегаты <code>Action<></code> и <code>Func<></code>	371
Понятие событий <code>C#</code>	373
Ключевое слово <code>event</code>	375
“За кулисами” событий	376
Прослушивание входящих событий	377
Упрощенная регистрация событий с использованием <code>Visual Studio</code>	378
Создание специальных аргументов событий	379
Обобщенный делегат <code>EventHandler<T></code>	381
Понятие анонимных методов <code>C#</code>	381
Доступ к локальным переменным	383
Понятие лямбда-выражений	384
Анализ лямбда-выражения	386
Обработка аргументов внутри множества операторов	387
Лямбда-выражения с несколькими параметрами и без параметров	388
Усовершенствование примера <code>PrimAndProperCarEvents</code> за счет использования лямбда-выражений	389
Резюме	390
Глава 11. Расширенные средства языка <code>C#</code>	391
Понятие методов-индексаторов	391
Индексация данных с использованием строковых значений	393
Перегрузка методов-индексаторов	394
Многомерные индексаторы	395
Определения индексаторов в интерфейсных типах	396

Понятие перегрузки операций	396
Перегрузка бинарных операций	397
А как насчет операций += и -=?	399
Перегрузка унарных операций	399
Перегрузка операций эквивалентности	400
Перегрузка операций сравнения	401
Финальные соображения относительно перегрузки операций	402
Понятие специальных преобразований типов	402
Числовые преобразования	402
Преобразования между связанными типами классов	403
Создание специальных процедур преобразования	403
Дополнительные явные преобразования для типа Square	406
Определение процедур неявного преобразования	406
Понятие расширяющих методов	408
Понятие анонимных типов	412
Определение анонимного типа	413
Анонимные типы, содержащие другие анонимные типы	417
Работа с типами указателей	417
Ключевое слово unsafe	419
Работа с операциями * и &	421
Небезопасная (и безопасная) функция обмена	421
Доступ к полям через указатели (операция ->)	422
Ключевое слово stackalloc	423
Закрепление типа с помощью ключевого слова fixed	423
Ключевое слово sizeof	424
Резюме	425
Глава 12. LINQ to Objects	426
Программные конструкции, специфичные для LINQ	426
Неявная типизация локальных переменных	427
Синтаксис инициализации объектов и коллекций	427
Лямбда-выражения	428
Расширяющие методы	429
Анонимные типы	429
Роль LINQ	430
Выражения LINQ строго типизированы	431
Основные сборки LINQ	431
Применение запросов LINQ к элементарным массивам	432
Решение без использования LINQ	433
Рефлексия результирующего набора LINQ	434
LINQ и неявно типизированные локальные переменные	435
LINQ и расширяющие методы	436
Роль отложенного выполнения	437
Роль немедленного выполнения	437
Возврат результата запроса LINQ	438
Возврат результатов LINQ через немедленное выполнение	440
Применение запросов LINQ к объектам коллекций	440
Доступ к содержащимся в контейнере подобъектам	441
Применение запросов LINQ к необобщенным коллекциям	442
Фильтрация данных с использованием OfType<T>()	442
Исследование операций запросов LINQ	443
Базовый синтаксис выборки	444
Получение подмножества данных	445

Проецирование новых типов данных	446
Получение счетчиков посредством Enumerable	447
Обращение результирующих наборов	447
Выражения сортировки	447
LINQ как лучшее средство построения диаграмм	448
Исключение дубликатов	449
Агрегатные операции LINQ	449
Внутреннее представление операторов запросов LINQ	450
Построение выражений запросов с использованием операций запросов	451
Построение выражений запросов с использованием типа Enumerable и лямбда-выражений	451
Построение выражений запросов с использованием типа Enumerable и анонимных методов	453
Построение выражений запросов с использованием типа Enumerable и низкоуровневых делегатов	453
Резюме	454
Глава 13. Время жизни объектов	455
Классы, объекты и ссылки	455
Базовые сведения о времени жизни объектов	456
Код CIL для ключевого слова new	457
Установка объектных ссылок в null	458
Роль корневых элементов приложения	459
Понятие поколений объектов	461
Параллельная сборка мусора в .NET 1.0 — .NET 3.5	462
Фоновая сборка мусора в .NET 4.0 и последующих версиях	462
Тип System.GC	463
Принудительный запуск сборщика мусора	464
Построение финализируемых объектов	466
Переопределение System.Object.Finalize()	467
Описание процесса финализации	469
Создание освобождаемых объектов	470
Повторное использование ключевого слова using в C#	472
Создание финализируемых и освобождаемых типов	473
Формализованный шаблон освобождения	474
Ленивое создание объектов	476
Настройка процесса создания данных Lazy<>	479
Резюме	480
Часть V. Программирование с использованием сборок .NET	481
Глава 14. Построение и конфигурирование библиотек классов	482
Определение специальных пространств имен	482
Устранение конфликтов между именами посредством полностью заданных имен	484
Устранение конфликтов между именами посредством псевдонимов	485
Создание вложенных пространств имен	487
Стандартное пространство имен Visual Studio	488
Роль сборки .NET	488
Сборки увеличивают возможности для повторного использования кода	489
Сборки определяют границы типов	489
Сборки являются единицами, поддерживающими версии	489
Сборки являются самоописательными	490
Сборки являются конфигурируемыми	490

Формат сборки .NET	490
Заголовок файла Windows	490
Заголовок файла CLR	492
CIL-код, метаданные типов и манифест сборки	493
Необязательные ресурсы сборки	494
Построение и использование специальной библиотеки классов	494
Исследование манифеста	497
Исследование CIL-кода	499
Исследование метаданных типов	500
Построение клиентского приложения на C#	500
Построение клиентского приложения на Visual Basic	502
Межязыковое наследование в действии	503
Понятие закрытых сборок	504
Идентичность закрытой сборки	504
Понятие процесса зондирования	504
Конфигурирование закрытых сборок	505
Роль файла App.Config	507
Понятие разделяемых сборок	508
Глобальный кеш сборок	509
Понятие строгих имен	510
Генерация строгих имен в командной строке	511
Генерация строгих имен в Visual Studio	513
Установка строго именованных сборок в GAC	514
Использование разделяемой сборки	515
Исследование манифеста SharedCarLibClient	517
Конфигурирование разделяемых сборок	517
Фиксация текущей версии разделяемой сборки	518
Построение разделяемой сборки версии 2.0.0.0	518
Динамическое перенаправление на специфичные версии разделяемой сборки	520
Понятие сборок политик издателя	522
Отключение политики издателя	523
Элемент <codeBase>	523
Пространство имен System.Configuration	525
Документация по схеме конфигурационного файла	526
Резюме	526
Глава 15. Рефлексия типов, позднее связывание и программирование с использованием атрибутов	528
Потребность в метаданных типов	528
Просмотр (частичных) метаданных для перечисления EngineState	529
Просмотр (частичных) метаданных для типа Car	530
Исследование блока TypeRef	531
Документирование определяемой сборки	532
Документирование ссылаемых сборок	532
Документирование строковых литералов	532
Понятие рефлексии	533
Класс System.Type	534
Получение информации о типе с помощью System.Object.GetType()	534
Получение информации о типе с помощью typeof()	535
Получение информации о типе с помощью System.Type.GetType()	535
Построение специального средства для просмотра метаданных	536
Рефлексия методов	536

Рефлексия полей и свойств	537
Рефлексия реализованных интерфейсов	537
Отображение различных дополнительных деталей	538
Реализация метода <code>Main()</code>	538
Рефлексия обобщенных типов	540
Рефлексия параметров и возвращаемых значений методов	540
Динамически загружаемые сборки	541
Рефлексия разделяемых сборок	543
Позднее связывание	545
Класс <code>System.Activator</code>	546
Вызов методов без параметров	547
Вызов методов с параметрами	548
Роль атрибутов .NET	549
Потребители атрибутов	550
Применение атрибутов в C#	550
Сокращенная нотация атрибутов C#	551
Указание параметров конструктора для атрибутов	552
Атрибут <code>[Obsolete]</code> в действии	552
Построение специальных атрибутов	553
Применение специальных атрибутов	553
Синтаксис именованных свойств	554
Ограничение использования атрибутов	554
Атрибуты уровня сборки	555
Файл <code>AssemblyInfo.cs</code> , генерируемый Visual Studio	556
Рефлексия атрибутов с использованием раннего связывания	557
Рефлексия атрибутов с использованием позднего связывания	558
Возможное применение на практике рефлексии, позднего связывания и специальных атрибутов	559
Построение расширяемого приложения	560
Построение сборки <code>CommonSnappableTypes.dll</code>	561
Построение оснастки на C#	561
Построение оснастки на Visual Basic	562
Построение расширяемого приложения Windows Forms	563
Резюме	566
Глава 16. Динамические типы и среда DLR	567
Роль ключевого слова <code>dynamic</code> языка C#	567
Вызов членов на динамически объявленных данных	569
Роль сборки <code>Microsoft.CSharp.dll</code>	570
Область применения ключевого слова <code>dynamic</code>	571
Ограничения ключевого слова <code>dynamic</code>	572
Практическое применение ключевого слова <code>dynamic</code>	572
Роль исполняющей среды динамического языка (DLR)	573
Роль деревьев выражений	574
Роль пространства имен <code>System.Dynamic</code>	574
Динамический поиск в деревьях выражений во время выполнения	575
Упрощение вызовов с поздним связыванием посредством динамических типов	575
Использование ключевого слова <code>dynamic</code> для передачи аргументов	576
Упрощение взаимодействия с COM посредством динамических данных	578
Роль основных сборок взаимодействия	579
Встраивание метаданных взаимодействия	580
Общие сложности взаимодействия с COM	581

Взаимодействие с COM с использованием динамических данных C#	582
Взаимодействие с COM без использования динамических данных C#	585
Резюме	586
Глава 17. Процессы, домены приложений и объектные контексты	588
Роль процесса Windows	588
Роль потоков	589
Взаимодействие с процессами на платформе .NET	591
Перечисление выполняющихся процессов	593
Исследование конкретного процесса	594
Исследование набора потоков процесса	594
Исследование набора модулей процесса	596
Запуск и останов процессов программным образом	597
Управление запуском процесса с использованием класса <code>ProcessStartInfo</code>	598
Домены приложений .NET	599
Класс <code>System.AppDomain</code>	600
Взаимодействие со стандартным доменом приложения	601
Перечисление загруженных сборок	602
Получение уведомлений о загрузке сборки	604
Создание новых доменов приложений	604
Загрузка сборки в специальные домены приложений	606
Выгрузка доменов приложений программным образом	607
Контекстные границы объектов	608
Контекстно-свободные и контекстно-связанные типы	609
Определение контекстно-связанного объекта	609
Исследование контекста объекта	609
Итоговые сведения о процессах, доменах приложений и контекстах	611
Резюме	611
Глава 18. Язык CIL и роль динамических сборок	612
Причины для изучения грамматики языка CIL	612
Директивы, атрибуты и коды операций CIL	613
Роль директив CIL	614
Роль атрибутов CIL	614
Роль кодов операций CIL	614
Разница между кодами операций и их мнемоническими эквивалентами в CIL	615
Основанная на стеке природа CIL	615
Возвратное проектирование	617
Роль меток в коде CIL	620
Взаимодействие с CIL: модификация файла <code>*.il</code>	620
Компиляция CIL-кода с помощью <code>ilasm.exe</code>	622
Роль инструмента <code>peverify.exe</code>	623
Директивы и атрибуты CIL	623
Указание ссылок на внешние сборки в CIL	623
Определение текущей сборки в CIL	624
Определение пространств имен в CIL	624
Определение типов классов в CIL	625
Определение и реализация интерфейсов в CIL	626
Определение структур в CIL	627
Определение перечислений в CIL	627
Определение обобщений в CIL	627
Компиляция файла <code>CILTypes.il</code>	628

Соответствия между базовыми классами .NET, C# и CIL	629
Определение членов типов в CIL	629
Определение полей данных в CIL	630
Определение конструкторов типа в CIL	630
Определение свойств в CIL	631
Определение параметров членов	631
Исследование кодов операций CIL	632
Директива <code>.maxstack</code>	634
Объявление локальных переменных в CIL	634
Отображение параметров на локальные переменные в CIL	635
Скрытая ссылка <code>this</code>	636
Представление итерационных конструкций в CIL	636
Создание сборки .NET на CIL	637
Построение сборки <code>CILCars.dll</code>	637
Построение сборки <code>CILCarClient.exe</code>	639
Динамические сборки	641
Исследование пространства имен <code>System.Reflection.Emit</code>	642
Роль типа <code>System.Reflection.Emit.ILGenerator</code>	643
Выдача динамической сборки	644
Выдача сборки и набора модулей	645
Роль типа <code>ModuleBuilder</code>	646
Выдача типа <code>HelloClass</code> и строковой переменной-члена	647
Выдача конструкторов	648
Выдача метода <code>SayHello()</code>	649
Использование динамически сгенерированной сборки	649
Резюме	650
Часть VI. Введение в библиотеки базовых классов .NET	651
Глава 19. Многопоточное, параллельное и асинхронное программирование	652
Отношения между процессом, доменом приложения, контекстом и потоком	653
Проблема параллелизма	654
Роль синхронизации потоков	654
Краткий обзор делегатов .NET	655
Асинхронная природа делегатов	656
Методы <code>BeginInvoke()</code> и <code>EndInvoke()</code>	657
Интерфейс <code>System.IAsyncResult</code>	657
Асинхронный вызов метода	658
Синхронизация вызывающего потока	658
Роль делегата <code>AsyncCallback</code>	660
Роль класса <code>AsyncResult</code>	662
Передача и получение специальных данных состояния	663
Пространство имен <code>System.Threading</code>	664
Класс <code>System.Threading.Thread</code>	664
Получение статистики о текущем потоке выполнения	666
Свойство <code>Name</code>	666
Свойство <code>Priority</code>	667
Ручное создание вторичных потоков	668
Работа с делегатом <code>ThreadStart</code>	668
Работа с делегатом <code>ParametrizedThreadStart</code>	670
Класс <code>AutoResetEvent</code>	671
Потоки переднего плана и фоновые потоки	672

Проблемы параллелизма	673
Синхронизация с использованием ключевого слова lock языка C#	675
Синхронизация с использованием типа System.Threading.Monitor	677
Синхронизация с использованием типа System.Threading.Interlocked	678
Синхронизация с использованием атрибута [Synchronization]	679
Программирование с использованием обратных вызовов Timer	680
Пул потоков CLR	681
Параллельное программирование с использованием TPL	683
Пространство имен System.Threading.Tasks	683
Роль класса Parallel	683
Обеспечение параллелизма данных с помощью класса Parallel	684
Доступ к элементам пользовательского интерфейса во вторичных потоках	686
Класс Task	688
Обработка запроса на отмену	688
Обеспечение параллелизма задач с помощью класса Parallel	689
Запросы Parallel LINQ (PLINQ)	692
Выполнение запроса PLINQ	693
Отмена запроса PLINQ	694
Асинхронные вызовы в версии .NET 4.5	695
Знакомство с ключевыми словами async и await языка C#	695
Соглашения об именовании асинхронных методов	697
Асинхронные методы, возвращающие void	698
Асинхронные методы с множеством контекстов await	699
Модернизация примера AddWithThreads с использованием async/await	699
Резюме	701
Глава 20. Файловый ввод-вывод и сериализация объектов	702
Исследование пространства имен System.IO	702
Классы Directory (DirectoryInfo) и File (FileInfo)	703
Абстрактный базовый класс FileSystemInfo	704
Работа с типом DirectoryInfo	705
Перечисление файлов с помощью типа DirectoryInfo	706
Создание подкаталогов с помощью типа DirectoryInfo	707
Работа с типом Directory	708
Работа с типом DriveInfo	709
Работа с классом FileInfo	710
Метод FileInfo.Create()	711
Метод FileInfo.Open()	711
Методы FileOpen.OpenRead() и FileInfo.OpenWrite()	712
Метод FileInfo.OpenText()	713
Методы FileInfo.CreateText() и FileInfo.AppendText()	713
Работа с типом File	714
Дополнительные члены File	714
Абстрактный класс Stream	715
Работа с классом FileStream	716
Работа с классами StreamWriter и StreamReader	717
Запись в текстовый файл	718
Чтение из текстового файла	719
Прямое создание экземпляров классов StreamWriter/StreamReader	720
Работа с классами StringWriter и StringReader	721
Работа с классами BinaryWriter и BinaryReader	722
Программное отслеживание файлов	723

Понятие сериализации объектов	725
Роль графов объектов	727
Конфигурирование объектов для сериализации	728
Определение сериализируемых типов	728
Открытые поля, закрытые поля и открытые свойства	729
Выбор формatera сериализации	730
Интерфейсы IFormatter и IRemotingFormatter	730
Точность типов среди форматов	731
Сериализация объектов с использованием BinaryFormatter	732
Десериализация объектов с использованием BinaryFormatter	733
Сериализация объектов с использованием SoapFormatter	734
Сериализация объектов с использованием XmlSerializer	735
Управление генерацией данных XML	736
Сериализация коллекций объектов	737
Настройка процессов сериализации SOAP и двоичной сериализации	739
Углубленный взгляд на сериализацию объектов	739
Настройка сериализации с использованием ISerializable	740
Настройка сериализации с использованием атрибутов	743
Резюме	744
Глава 21. ADO.NET, часть I: подключенный уровень	745
Высокоуровневое определение ADO.NET	745
Три грани ADO.NET	746
Поставщики данных ADO.NET	747
Поставщики данных ADO.NET от Microsoft	749
О сборке System.Data.OracleClient.dll	750
Получение сторонних поставщиков данных ADO.NET	750
Дополнительные пространства имен ADO.NET	750
Типы из пространства имен System.Data	751
Роль интерфейса IDbConnection	752
Роль интерфейса IDbTransaction	752
Роль интерфейса IDbCommand	753
Роль интерфейсов IDbDataParameter и IDataParameter	753
Роль интерфейсов IDbDataAdapter и IDataAdapter	754
Роль интерфейсов IDataReader и IDataRecord	754
Абстрагирование поставщиков данных с помощью интерфейсов	755
Повышение гибкости с помощью конфигурационных файлов приложения	757
Создание базы данных AutoLot	758
Создание таблицы Inventory	758
Занесение тестовых записей в таблицу Inventory	760
Создание хранимой процедуры GetPetName()	761
Создание таблиц Customers и Orders	761
Визуальное создание отношений между таблицами	763
Модель фабрик поставщиков данных ADO.NET	764
Полный пример фабрики поставщиков данных	765
Потенциальный недостаток модели фабрик поставщиков данных	767
Элемент <connectionStrings>	768
Понятие подключенного уровня ADO.NET	769
Работа с объектами подключения	770
Работа с объектами ConnectionStringBuilder	772
Работа с объектами команд	773

Работа с объектами чтения данных	774
Получение нескольких результирующих наборов с использованием объекта чтения данных	776
Создание многократно используемой библиотеки доступа к данным	776
Добавление логики подключения	778
Добавление логики вставки	778
Добавление логики удаления	779
Добавление логики обновления	780
Добавление логики выборки	780
Работа с параметризованными объектами команд	781
Выполнение хранимой процедуры	783
Создание приложения с консольным пользовательским интерфейсом	785
Реализация метода Main()	785
Реализация метода ShowInstructions()	787
Реализация метода ListInventory()	787
Реализация метода DeleteCar()	788
Реализация метода InsertNewCar()	788
Реализация метода UpdateCarPetName()	789
Реализация метода LookUpPetName()	789
Понятие транзакций базы данных	790
Основные члены объекта транзакции ADO.NET	791
Добавление таблицы CreditRisks в базу данных AutoLot	792
Добавление метода транзакции в InventoryDAL	792
Тестирование транзакции базы данных	793
Резюме	794
Глава 22. ADO.NET, часть II: автономный уровень	795
Понятие автономного уровня ADO.NET	795
Роль объектов DataSet	797
Основные свойства класса DataSet	797
Основные методы класса DataSet	797
Создание DataSet	798
Работа с объектами DataColumn	799
Построение объекта DataColumn	800
Включение автоинкрементных полей	801
Добавление объектов DataColumn в DataTable	801
Работа с объектами DataRow	802
Свойство RowState	803
Свойство DataRowVersion	804
Работа с объектами DataTable	805
Вставка объектов DataTable в DataSet	806
Получение данных из объекта DataSet	806
Обработка данных из DataTable с использованием объектов DataTableReader	807
Сериализация объектов DataTable и DataSet в формате XML	808
Сериализация объектов DataTable и DataSet в двоичном формате	810
Привязка объектов DataTable к графическим пользовательским интерфейсам	
Windows Forms	811
Заполнение DataTable из обобщенного List<T>	812
Удаление строк из DataTable	814
Выборка строк на основе критерия фильтрации	815
Обновление строк в DataTable	817
Работа с типом DataView	818

Работа с адаптерами данных	819
Простой пример адаптера данных	820
Отображение имен из базы данных на дружественные к пользователю имена	821
Добавление в AutoLotDAL.dll функциональности автономного уровня	822
Определение начального класса	822
Конфигурирование адаптера данных с использованием SqlCommandBuilder	823
Реализация метода GetAllInventory()	824
Реализация метода UpdateInventory()	824
Установка номера версии	825
Тестирование функциональности автономного уровня	825
Объекты DataSet с несколькими таблицами и отношения между данными	826
Подготовка адаптеров данных	827
Построение отношений между таблицами	828
Обновление таблиц базы данных	829
Навигация между связанными таблицами	829
Инструменты визуального конструирования баз данных Windows Forms	831
Визуальное проектирование элемента управления DataGridView	832
Сгенерированный файл App.config	835
Исследование строго типизированного класса DataSet	835
Исследование строго типизированного класса DataTable	836
Исследование строго типизированного класса DataRow	837
Исследование строго типизированного адаптера данных	837
Завершение приложения Windows Forms	838
Изоляция строго типизированного кода работы с базой данных в библиотеке классов	839
Просмотр сгенерированного кода	840
Выборка данных с помощью сгенерированного кода	841
Вставка данных с помощью сгенерированного кода	842
Удаление данных с помощью сгенерированного кода	843
Вызов хранимой процедуры с помощью сгенерированного кода	843
Программирование с помощью LINQ to DataSet	844
Роль библиотеки расширений DataSet	846
Получение объекта DataTable, совместимого с LINQ	846
Роль расширяющего метода DataRowExtensions.Field<T>()	848
Заполнение новых объектов DataTable из запросов LINQ	848
Резюме	849
Глава 23. ADO.NET, часть III: Entity Framework	850
Роль Entity Framework	850
Роль сущностей	852
Строительные блоки Entity Framework	854
Роль служб объектов	854
Роль клиента сущности	855
Роль файла *.edmx	856
Роль классовObjectContext и ObjectSet<T>	857
Собираем все вместе	858
Построение и анализ первой модели EDM	859
Генерация файла *.edmx	859
Изменение формы сущностных данных	862
Просмотр отображений	864
Просмотр сгенерированного файла *.edmx	865
Просмотр сгенерированного исходного кода	867
Улучшение сгенерированного исходного кода	869

Программирование с использованием концептуальной модели	869
Удаление записи	870
Обновление записи	871
Запросы с помощью LINQ to Entities	872
Запросы с помощью Entity SQL	873
Работа с объектом EntityDataReader	874
Проект AutoLotDAL версии 4, теперь с сущностями	875
Роль навигационных свойств	875
Использование навигационных свойств внутри запросов LINQ to Entity	877
Вызов хранимой процедуры	878
Привязка данных сущностей к графическим пользовательским интерфейсам	
Windows Forms	879
Добавление кода привязки данных	882
Продолжение изучения API-интерфейсов доступа к данным в .NET	882
Резюме	883
Глава 24. Введение в LINQ to XML	884
История о двух API-интерфейсах XML	884
Интерфейс LINQ to XML как лучшая модель DOM	886
Синтаксис литералов Visual Basic как наилучший интерфейс LINQ to XML	886
Члены пространства имен System.Xml.Linq	887
Осевые методы LINQ to XML	889
Избыточность XName (и XNamespace)	890
Работа с XElement и XDocument	891
Генерация документов из массивов и контейнеров	893
Загрузка и разбор XML-содержимого	894
Манипулирование XML-документом в памяти	894
Построение пользовательского интерфейса приложения LINQ to XML	895
Импорт файла Inventory.xml	895
Определение вспомогательного класса LINQ to XML	896
Связывание пользовательского интерфейса и вспомогательного класса	897
Резюме	898
Глава 25. Введение в Windows Communication Foundation	899
Собрание API-интерфейсов распределенных вычислений	899
Роль DCOM	900
Роль служб COM+/Enterprise Services	901
Роль MSMQ	902
Роль .NET Remoting	902
Роль веб-служб XML	903
Роль WCF	904
Обзор функциональных возможностей WCF	905
Обзор архитектуры, ориентированной на службы	905
WCF: заключение	906
Исследование основных сборок WCF	907
Шаблоны проектов WCF в Visual Studio	908
Шаблон проекта для построения веб-сайта со службами WCF	909
Базовая структура приложения WCF	909
Адрес, привязка и контракт в WCF	911
Понятие контрактов WCF	911
Понятие привязок WCF	912
Понятие адресов WCF	914

Построение службы WCF	916
Атрибут [ServiceContract]	917
Атрибут [OperationContract]	918
Служебные типы как контракты операций	918
Хостинг службы WCF	919
Установка ABC внутри файла App.config	920
Кодирование с использованием типа ServiceHost	921
Указание базовых адресов	921
Подробный анализ типа ServiceHost	922
Подробный анализ элемента <system.serviceModel>	924
Включение обмена метаданными	924
Построение клиентского приложения WCF	927
Генерация кода прокси с использованием svcutil.exe	927
Генерация кода прокси с использованием Visual Studio	928
Конфигурирование привязки на основе TCP	930
Упрощение конфигурационных настроек	931
Использование стандартных конечных точек	931
Открытие одной службы WCF, использующей множество привязок	932
Изменение параметров привязки WCF	934
Использование конфигурации стандартного поведения MEX	935
Обновление клиентского прокси и выбор привязки	936
Использование шаблона проекта WCF Service Library	937
Построение простой математической службы	938
Тестирование службы WCF с помощью WcfTestClient.exe	938
Изменение конфигурационных файлов с помощью SvcConfigEditor.exe	939
Хостинг службы WCF внутри Windows-службы	940
Указание ABC в коде	941
Включение MEX	943
Создание программы установки для Windows-службы	943
Установка Windows-службы	944
Асинхронный вызов службы из клиента	945
Проектирование контрактов данных WCF	947
Использование веб-ориентированного шаблона проекта WCF Service	948
Реализация контракта службы	949
Роль файла *.svc	951
Содержимое файла Web.config	951
Тестирование службы	951
Резюме	952
Глава 26. Введение в Windows Workflow Foundation	953
Определение бизнес-процесса	953
Роль WF	954
Построение простого рабочего потока	954
Исполняющая среда рабочих потоков	957
Хостинг рабочего потока с использованием класса WorkflowInvoker	957
Хостинг рабочего потока с использованием класса WorkflowApplication	960
Итоги по первому рабочему потоку	961
Знакомство с действиями рабочих потоков	961
Действия потока управления	962
Действия блок-схемы	962
Действия обмена сообщениями	963
Действия конечного автомата	963
Действия исполняющей среды и действия-примитивы	964

Действия транзакций	964
Действия над коллекциями и действия обработки ошибок	965
Построение рабочего потока в виде блок-схемы	965
Подключение действий к блок-схеме	966
Работа с действием InvokeMethod	967
Определение переменных уровня рабочего потока	968
Работа с действием FlowDecision	969
Работа с действием TerminateWorkflow	970
Построение условия True	970
Работа с действием ForEach<T>	971
Завершение приложения	973
Промежуточные итоги	974
Построение последовательного рабочего потока (в выделенной библиотеке)	975
Определение начального проекта	975
Импорт сборок и пространств имен	976
Определение аргументов рабочего потока	977
Определение переменных рабочего потока	978
Работа с действием Assign	979
Работа с действиями If и Switch	980
Построение специального действия кода	982
Использование библиотеки рабочего потока	984
Извлечение выходного аргумента рабочего потока	985
Резюме	986
Часть VII. Windows Presentation Foundation	987
Глава 27. Введение в Windows Presentation Foundation и XAML	988
Мотивация, лежащая в основе WPF	988
Унификация различных API-интерфейсов	989
Обеспечение разделения ответственности через XAML	990
Обеспечение оптимизированной модели визуализации	990
Упрощение программирования сложных пользовательских интерфейсов	991
Различные варианты приложений WPF	992
Традиционные настольные приложения	992
WPF-приложения на основе навигации	994
Приложения XBAP	994
Отношения между WPF и Silverlight	995
Исследование сборки WPF	996
Роль класса Application	998
Роль класса Window	999
Роль класса System.Windows.Controls.ContentControl	999
Роль класса System.Windows.Controls.Control	1001
Роль класса System.Windows.FrameworkElement	1002
Роль класса System.Windows.UIElement	1002
Роль класса System.Windows.Media.Visual	1003
Роль класса System.Windows.DependencyObject	1003
Роль класса System.Windows.Threading.DispatcherObject	1003
Построение приложения WPF без XAML	1003
Создание строго типизированного окна	1005
Создание простого пользовательского интерфейса	1006
Взаимодействие с данными уровня приложения	1007
Обработка закрытия объекта Window	1008
Перехват событий мыши	1009
Перехват клавиатурных событий	1010

Построение приложения WPF с использованием только XAML	1011
Определение объекта Window в XAML	1012
Определение объекта Application в XAML	1014
Обработка файлов XAML с помощью msbuild.exe	1014
Трансформация разметки в сборку .NET	1016
Отображение XAML-разметки окна на код C#	1016
Роль BAML	1018
Отображение XAML-разметки приложения на код C#	1019
Итоговые замечания о процессе трансформирования XAML в сборку	1020
Синтаксис XAML для WPF	1020
Введение в Xaml	1020
Пространства имен XAML XML и “ключевые слова” XAML	1022
Управление видимостью классов и переменных-членов	1024
Элементы XAML, атрибуты XAML и преобразователи типов	1024
Понятие синтаксиса “свойство-элемент” в XAML	1026
Понятие присоединяемых свойств XAML	1026
Понятие расширений разметки XAML	1027
Построение приложений WPF с использованием файлов отделенного кода	1029
Добавление файла кода для класса MainWindow	1029
Добавление файла кода для класса MyApp	1030
Обработка файлов кода с помощью msbuild.exe	1031
Построение приложений WPF с использованием Visual Studio	1031
Шаблоны проектов WPF	1032
Панель инструментов и визуальный конструктор/редактор XAML	1032
Установка свойств с использованием окна Properties	1034
Обработка событий с использованием окна Properties	1034
Обработка событий в редакторе XAML	1035
Окно Document Outline	1036
Просмотр автоматически сгенерированных файлов кода	1036
Построение специального редактора XAML с помощью Visual Studio	1036
Проектирование графического пользовательского интерфейса окна	1037
Реализация события Loaded	1038
Реализация события Click объекта Button	1039
Реализация события Closed	1040
Тестирование приложения	1040
Изучение документации WPF	1041
Резюме	1042
Глава 28. Программирование с использованием элементов управления WPF	1043
Обзор основных элементов управления WPF	1043
Элементы управления Ink API	1044
Элементы управления документов WPF	1045
Общие диалоговые окна WPF	1045
Подробные сведения находятся в документации	1045
Краткий обзор визуального конструктора WPF в Visual Studio	1046
Работа с элементами управления WPF в Visual Studio	1047
Работа с редактором Document Outline	1047
Управление компоновкой содержимого с использованием панелей	1048
Позиционирование содержимого внутри панелей Canvas	1051
Позиционирование содержимого внутри панелей WrapPanel	1052
Позиционирование содержимого внутри панелей StackPanel	1053
Позиционирование содержимого внутри панелей Grid	1054
Позиционирование содержимого внутри панелей DockPanel	1057

Включение прокрутки в типах панелей	1058
Конфигурирование панелей с использованием визуальных конструкторов Visual Studio	1058
Построение окна с использованием вложенных панелей	1061
Построение системы меню	1062
Построение панели инструментов	1065
Построение строки состояния	1066
Завершение проектирования пользовательского интерфейса	1066
Реализация обработчиков событий MouseEnter/MouseLeave	1067
Реализация логики проверки правописания	1067
Понятие команд WPF	1068
Объекты внутренних команд	1068
Подключение команд к свойству Command	1069
Подключение команд к произвольным действиям	1070
Работа с командами Open и Save	1071
Более глубокий взгляд на API-интерфейсы и элементы управления WPF	1073
Работа с элементом управления TabControl	1074
Построение вкладки Ink API	1075
Проектирование панели инструментов	1076
Элемент управления RadioButton	1078
Обработка событий для вкладки Ink API	1078
Элемент управления InkCanvas	1080
Элемент управления ComboBox	1082
Сохранение, загрузка и очистка данных InkCanvas	1084
Введение в интерфейс Documents API	1085
Блочные элементы и встроенные элементы	1085
Диспетчеры компоновки документа	1085
Построение вкладки Documents	1086
Наполнение FlowDocument с помощью кода	1087
Включение аннотаций и “клеяких” заметок	1088
Сохранение и загрузка потокового документа	1089
Введение в модель привязки данных WPF	1091
Построение вкладки Data Binding	1091
Установка привязки данных с использованием Visual Studio	1092
Свойство DataContext	1093
Преобразование данных с использованием IValueConverter	1094
Установка привязок данных в коде	1095
Построение вкладки DataGrid	1096
Резюме	1097
Глава 29. Службы визуализации графики WPF	1098
Службы графической визуализации WPF	1098
Варианты графической визуализации WPF	1099
Визуализация графических данных с использованием фигур	1100
Добавление прямоугольников, эллипсов и линий на поверхность Canvas	1102
Удаление прямоугольников, эллипсов и линий с поверхности Canvas	1105
Работа с элементами Polyline и Polygon	1106
Работа с элементом Path	1107
Кисти и перья WPF	1110
Конфигурирование кистей с использованием Visual Studio	1111
Конфигурирование кистей в коде	1112
Конфигурирование перьев	1113

Применение графических трансформаций	1114
Первый взгляд на трансформации	1115
Трансформация данных Canvas	1115
Работа с редактором трансформаций Visual Studio	1118
Построение начальной компоновки	1118
Применение трансформаций на этапе проектирования	1119
Трансформация холста в коде	1120
Визуализация графических данных с использованием рисунков и геометрий	1121
Построение кисти DrawingBrush с использованием геометрий	1122
Рисование с помощью DrawingBrush	1123
Включение типов Drawing в DrawingImage	1124
Роль инструмента Expression Design	1125
Экспорт файла с примером графики в виде XAML	1125
Импорт графических данных в проект WPF	1127
Взаимодействие с объектами изображения	1128
Визуализация графических данных с использованием визуального уровня	1129
Базовый класс Visual и производные дочерние классы	1129
Первый взгляд на класс DrawingVisual	1130
Визуализация графических данных в специальном диспетчере компоновки	1132
Реагирование на операции проверки попадания	1133
Резюме	1135
Глава 30. Ресурсы, анимация и стили WPF	1136
Система ресурсов WPF	1136
Работа с двоичными ресурсами	1137
Программная загрузка изображения	1139
Работа с объектными (логическими) ресурсами	1142
Роль свойства Resources	1142
Определение ресурсов уровня окна	1142
Расширение разметки {StaticResource}	1145
Расширение разметки {DynamicResource}	1145
Ресурсы уровня приложения	1146
Определение объединенных словарей ресурсов	1147
Определение сборки, включающей только ресурсы	1148
Службы анимации WPF	1150
Роль классов анимации	1150
Свойства To, From и By	1151
Роль базового класса Timeline	1152
Реализация анимации в коде C#	1152
Управление темпом анимации	1153
Запуск в обратном порядке и циклическое выполнение анимации	1154
Реализация анимации в разметке XAML	1155
Роль раскадровок	1156
Роль триггеров событий	1156
Анимация с использованием дискретных ключевых кадров	1157
Роль стилей WPF	1158
Определение и применение стиля	1158
Переопределение настроек стиля	1159
Автоматическое применение стиля с помощью TargetType	1159
Создание подклассов существующих стилей	1160
Роль неименованных стилей	1160
Определение стилей с триггерами	1161
Определение стилей с несколькими триггерами	1162

Анимированные стили	1162
Применение стилей в коде	1163
Резюме	1164
Глава 31. Свойства зависимости, маршрутизируемые события и шаблоны	1165
Роль свойств зависимости	1165
Знакомство с существующим свойством зависимости	1167
Важные замечания относительно оболочек свойств CLR	1170
Построение специального свойства зависимости	1170
Добавление процедуры проверки достоверности данных	1174
Реагирование на изменение свойства	1175
Маршрутизируемые события	1176
Роль маршрутизируемых пузырьковых событий	1177
Продолжение или прекращение пузырькового распространения	1177
Роль маршрутизируемых туннельных событий	1178
Логические деревья, визуальные деревья и стандартные шаблоны	1180
Программный просмотр логического дерева	1180
Программный просмотр визуального дерева	1181
Программный просмотр стандартного шаблона элемента управления	1183
Построение специального шаблона элемента управления с помощью инфраструктуры триггеров	1186
Шаблоны как ресурсы	1187
Встраивание визуальных подсказок с использованием триггеров	1188
Роль расширения разметки {TemplateBinding}	1189
Роль класса ContentPresenter	1190
Включение шаблонов в стили	1191
Резюме	1192
Часть VIII. ASP.NET Web Forms	1193
Глава 32. Введение в ASP.NET Web Forms	1194
Роль протокола HTTP	1194
Цикл запрос/ответ HTTP	1195
HTTP — протокол без хранения состояния	1195
Веб-приложения и веб-серверы	1195
Роль виртуальных каталогов IIS	1196
Веб-сервер разработки ASP.NET	1197
Роль языка HTML	1197
Структура HTML-документа	1198
Роль форм HTML	1199
Инструменты визуального конструктора HTML в Visual Studio	1199
Построение HTML-формы	1200
Роль сценариев клиентской стороны	1202
Пример сценария клиентской стороны	1203
Обратная отправка веб-серверу	1204
Обратные отправки в ASP.NET	1205
Обзор API-интерфейса ASP.NET	1205
Основные функциональные возможности ASP.NET 2.0 и последующих версий	1206
Основные функциональные возможности ASP.NET 3.5 (и .NET 3.5 SP1) и последующих версий	1207
Основные функциональные возможности ASP.NET 4.0 и 4.5	1208
Построение однофайловой веб-страницы ASP.NET	1208
Указание сборки AutoLotDAL.dll	1210

Проектирование пользовательского интерфейса	1210
Добавление логики доступа к данным	1211
Роль директив ASP.NET	1213
Анализ блока script	1214
Анализ объявлений элементов управления ASP.NET	1215
Построение веб-страницы ASP.NET с использованием файлов кода	1216
Ссылка на сборку AutoLotDAL.dll	1218
Изменение файла кода	1218
Отладка и трассировка страниц ASP.NET	1219
Сравнение веб-сайтов и веб-приложений ASP.NET	1220
Структура каталогов веб-сайта ASP.NET	1222
Ссылка на сборки	1222
Роль папки App_Code	1223
Цепочка наследования для типа Page	1223
Взаимодействие с входящим HTTP-запросом	1225
Получение статистики о браузере	1226
Доступ к входным данным формы	1227
Свойство IsPostBack	1227
Взаимодействие с исходящим HTTP-ответом	1228
Выдача HTML-содержимого	1229
Перенаправление пользователей	1229
Жизненный цикл веб-страницы ASP.NET	1230
Роль атрибута AutoEventWireup	1231
Событие Error	1231
Роль файла web.config	1233
Утилита администрирования веб-сайтов ASP.NET	1234
Резюме	1234
Глава 33. Веб-элементы управления, мастер-страницы и темы ASP.NET	1235
Природа веб-элементов управления	1235
Обработка событий серверной стороны	1236
Свойство AutoPostBack	1237
Базовые классы Control и WebControl	1238
Перечисление содержащихся элементов управления	1238
Динамическое добавление и удаление элементов управления	1241
Взаимодействие с динамически созданными элементами управления	1242
Функциональность базового класса WebControl	1243
Основные категории веб-элементов управления ASP.NET	1243
Несколько слов о пространстве имен System.Web.UI.HtmlControls	1245
Документация по веб-элементам управления	1245
Построение примера веб-сайта ASP.NET	1245
Работа с мастер-страницами	1246
Определение стандартной страницы содержимого Default.aspx	1252
Проектирование страницы содержимого Inventory.aspx	1254
Проектирование страницы содержимого BuildCar.aspx	1258
Роль элементов управления проверкой достоверности	1261
Включение поддержки проверки достоверности с помощью JavaScript на стороне клиента	1263
Элемент управления RequiredFieldValidator	1263
Элемент управления RegularExpressionValidator	1264
Элемент управления RangeValidator	1264
Элемент управления CompareValidator	1264

Создание итоговой панели проверки достоверности	1265
Определение групп проверки достоверности	1266
Работа с темами	1268
Файлы *.skin	1269
Применение тем ко всему сайту	1271
Применение тем на уровне страницы	1271
Свойство SkinID	1271
Программное назначение тем	1272
Резюме	1273
Глава 34. Управление состоянием в ASP.NET	1274
Проблема поддержки состояния	1274
Приемы управления состоянием ASP.NET	1276
Роль состояния представления ASP.NET	1277
Демонстрация работы с состоянием представления	1277
Добавление специальных данных в состояние представления	1279
Роль файла Global.asax	1279
Глобальный обработчик исключений “последнего шанса”	1281
Базовый класс <code>HttpApplication</code>	1282
Различие между состоянием приложения и состоянием сеанса	1282
Поддержка данных состояния уровня приложения	1282
Модификация данных приложения	1285
Обработка останова веб-приложения	1286
Работа с кешем приложения	1286
Использование кеширования данных	1287
Модификация файла *.aspx	1289
Поддержка данных сеанса	1291
Дополнительные члены класса <code>HttpSessionState</code>	1293
Cookie-наборы	1294
Создание cookie-наборов	1295
Чтение входящих cookie-данных	1296
Роль элемента <code><sessionState></code>	1297
Хранение данных сеанса на сервере состояния сеансов ASP.NET	1297
Хранение информации о сеансах в выделенной базе данных	1298
Введение в API-интерфейс ASP.NET Profile	1299
База данных <code>ASPNETDB.mdf</code>	1299
Определение пользовательского профиля в <code>web.config</code>	1300
Программный доступ к данным профиля	1301
Группирование данных профиля и сохранение специальных объектов	1303
Резюме	1305
Предметный указатель	1306