Всему роду Троелсен: Мэри (матери), Уолтеру (отцу), Мэнди (жене) и Сорену (сыну). Нам не хватает тебя, Микко (кот).

Эндрю

Моей семье, Эми (жене), Коннеру (сыну), Логану (сыну) и Скайлер (дочери), спасибо за поддержку и терпение с вашей стороны.

Филипп

Введение

Первое издание этой книги вышло в 2001 году в то же самое время, когда компания Microsoft выпустила сборку Beta 2 платформы .NET 1.0. Работа над первым изданием, безусловно, была сложной задачей, поскольку в процессе написания книги API-интерфейсы и язык C# постоянно менялись. В то время полнота платформы .NET с точки зрения разработчика была вполне управляемой. Инфраструктура Windows Forms являлась единственным API-интерфейсом для построения графических пользовательских интерфейсов настольных приложений в рамках платформы, инфраструктура ASP.NET была сконцентрирована исключительно на модели программирования с веб-формами, а язык C# не содержал ничего лишнего помимо средств объектно-ориентированного программирования.

На протяжении первых шести изданий книги я был единственным автором, ответственным за обновление текста для учета многочисленных изменений в языке C# и новых API-интерфейсов внутри платформы .NET. Эта книга обновлялась в течение последних 14 лет, чтобы принять во внимание модель программирования Language Integrated Query (LINQ), Windows Presentation Foundation (WPF), Windows Communication Foundation (WCF), новые модели и ключевые слова, связанные с многопоточностью, новые инструменты для разработки, а также изменения в инфраструктуре программирования веб-приложений (не считая многих других аспектов).

Начиная с седьмого издания, мне стало ясно, что полное обновление книги потребовало бы исключительно долгого времени. Несомненно, моя собственная жизнь стала намного более занятой, чем было в 2001 году (и даже в 2011 году; я подозреваю, что причина в том, что я стал отцом, или, может быть, слишком большое число университетских курсов… трудно сказать). Так или иначе, но когда в Apress мне предложили обновить книгу с целью учета последнего выпуска платформы .NET, мы обдумали целый ряд разных подходов, чтобы получить книгу своевременно. В конечном счете, мы пришли к заключению, что обновление текста без посторонней помощи может занять значительное время. В Apress мне посоветовали встретиться с Филиппом Джепиксом, чтобы посмотреть, получится ли из этого хорошая партия. После нескольких телефонных разговоров, переписки по электронной почте и тщательных размышлений я был счастлив пригласить его присоединиться к проекту. Я рад сообщить, что настоящая книга является результатом совместных усилий меня самого и соавтора Филиппа Джепикса. Позвольте Филиппу представить себя самостоятельно…

Эндрю Троелсен

Займись этим, Филипп!

Когда компания Microsoft выпускала ранние бета-версии .NET, я уже плотно занимался технологией Microsoft. Я перешел с других технологий на Visual Basic и строил клиентские приложения, а также веб-сайты на классическом ASP с использованием VB и MTS. Хотя на то время эти инструменты работали хорошо, я предвидел их скорый конец.

Исследование других стеков технологий привело меня к первому изданию данной книги. Я прочитал ее от корки до корки и увидел не только перспективу .NET, но также важность этой книги в моей библиотеке технологий. Наличие одной книги, раскрывающей язык C# и всю экосистему .NET, было бесценным. Не могу точно сказать, сколько экземпляров настоящей книги продало издательство Apress на протяжении прошедших лет, но своим клиентам и слушателям я рекомендовал приобрести каждое издание в качестве первой книги по C# и .NET. Независимо от того, начинаете вы изучать .NET или просто интересуетесь нововведениями в последней версии платформы, трудно найти книгу лучше этой.

Я уже написал одну книгу для Apress, посвященную Windows 8.1 и C#, так что был знаком с командой и оценил методы, которыми издательство Apress содействует работе и поддерживает своих авторов. Когда мой редактор связался со мной и спросил, не хотел бы я совместно потрудиться над седьмым изданием "книги Троелсена" (так я ее всегда называл), я был взволнован и польщен. Это было по-настоящему любимой работой, и мне очень трудно выразить словами, насколько я счастлив считать себя частицей данной книги. Я надеюсь, что вы получите столько же удовольствия при чтении книги, сколько я получил во время ее написания и работы с Эндрю и потрясающей командой из Apress.

Филипп Джепикс

Авторы и читатели — одна команда

Авторам книг по технологиям приходится писать для очень требовательной группы людей. Вам известно, что построение программных решений с применением любой платформы или языка исключительно сложно и специфично для отдела, компании, клиентской базы и поставленной задачи. Возможно, вы работаете в индустрии электронных публикаций, разрабатываете системы для правительства или местных органов власти либо сотрудничаете с NASA или какой-то военной отраслью. Вместе мы трудимся в нескольких отраслях, включая разработку обучающего программного обеспечения для детей (Oregon Trail/Amazon Trail), разнообразных производственных систем и проектов в медицинской и финансовой сфере. Написанный вами код на месте вашего трудоустройства на 100% будет иметь мало общего с кодом, который создавали мы на протяжении многих лет.

По указанной причине в этой книге мы намеренно решили избегать демонстрации примеров кода, свойственного какой-то конкретной отрасли или направлению программирования. Таким образом, мы объясняем язык C#, объектно-ориентированное программирование, среду CLR и библиотеки базовых классов .NET с использованием примеров, не привязанных к отрасли. Вместо того чтобы заставлять каждый пример наполнять сетку данными, подчитывать фонд заработной платы или делать еще что-нибудь подобное, мы придерживаемся темы, близкой каждому из нас: автомобили (с добавлением умеренного количества геометрических структур и систем расчета заработной платы для сотрудников). И вот тут наступает ваш черед.

Наша работа заключается в как можно лучшем объяснении языка программирования C# и основных аспектов платформы .NET. Мы также будем делать все возможное для того, чтобы снарядить вас инструментами и стратегиями, которые вам необходимы для продолжения обучения после завершения работы с данной книгой.

Ваша работа предусматривает усвоение этой информации и ее применение к решению своих задач программирования. Мы полностью отдаем себе отчет, что ваши проекты, скорее всего, не будут связаны с автомобилями и их дружественными именами, но именно в этом состоит суть прикладных знаний.

Мы уверены, что после освоения тем и концепций, представленных в этой книге, вы сможете успешно строить решения .NET, которые соответствуют вашей конкретной среде программирования.

Краткий обзор книги

Книга логически разделена на восемь частей, каждая из которых содержит несколько связанных друг с другом глав. Ниже приведено краткое содержание частей и глав.

Часть I. Введение в C# и платформу .NET

Часть I этой книги предназначена для ознакомления с природой платформы .NET и различными инструментами разработки (включая межплатформенные IDE-среды), которые используются во время построения приложений .NET.

Глава 1. Философия .NET

Первая глава выступает в качестве основы для всего остального материала. Ее главная цель в том, чтобы представить вам набор строительных блоков .NET, таких как общеязыковая исполняющая среда (Common Language Runtime), общая система типов (Common Type System), общеязыковая спецификация (Common Language Specification) и библиотеки базовых классов. Здесь вы впервые взглянете на язык программирования C# и формат сборок .NET. Кроме того, будет исследована независимая от платформы природа .NET.

Глава 2. Создание приложений на языке C#

Целью этой главы является введение в процесс компиляции файлов исходного кода C# с применением разнообразных средств и приемов. Здесь вы узнаете о роли инструментов разработки Microsoft Express и совершенно бесплатного (но полнофункционального) продукта Visual Studio Community Edition, который используется повсеместно в книге. Мы также кратко коснемся роли IDE-среды Xamarin и покажем, как она делает возможной разработку приложений .NET в средах операционных систем Linux и Mac OS X. Вы также научитесь конфигурировать машину разработки с установленной локальной копией всей важной документации .NET Framework 4.6 SDK.

Часть II. Основы программирования на C#

Темы, представленные в этой части книги, очень важны, потому что они связаны с разработкой программного обеспечения .NET любого типа (например, веб-приложений, настольных приложений с графическим пользовательским интерфейсом, библиотек кода или служб Windows). Здесь вы узнаете о фундаментальных типах данных .NET, научитесь манипулировать текстом и ознакомитесь с ролью модификаторов параметров C# (включая необязательные и именованные аргументы).

Глава 3. Главные конструкции программирования на C#: часть I

В этой главе начинается формальное исследование языка программирования C#. Здесь рассматривается роль метода `Main()` и многочисленные детали, касающиеся внутренних типов данных .NET, среди которых манипулирование текстовыми данными с применением типов `System.String` и `System.Text.StringBuilder`. Кроме того, будут описаны итерационные конструкции и конструкции принятия решений, сужающие и расширяющие операции, а также ключевое слово `unchecked`.

Глава 4. Главные конструкции программирования на C#: часть II

В этой главе завершается исследование ключевых аспектов C#. Будет показано, каким образом конструировать перегруженные методы типов и определять параметры с использованием ключевых слов `out`, `ref` и `params`. Также рассматриваются два средства языка C#: именованные и необязательные параметры. Вы узнаете, как создавать

и манипулировать массивами данных, а также определять типы, допускающие `null` (с операциями ? и ??). Вдобавок вы освоите отличия между типами значений (в том числе перечисления и специальные структуры) и ссылочными типами.

Часть III. Объектно-ориентированное программирование на C#

В этой части вы изучите ключевые конструкции языка C#, включая детали объектно-ориентированного программирования. Здесь вы научитесь обрабатывать исключения времени выполнения и взаимодействовать со строго типизированными интерфейсами.

Глава 5. Инкапсуляция

В этой главе начинается рассмотрение концепций объектно-ориентированного программирования (ООП) на языке C#. После представления главных принципов ООП (инкапсуляции, наследования и полиморфизма) будет показано, как строить надежные типы классов с применением конструкторов, свойств, статических членов, констант и полей только для чтения. Глава завершается исследованием частичных определений типов, синтаксиса инициализации объектов и автоматических свойств.

Глава 6. Наследование и полиморфизм

Здесь вы ознакомитесь с оставшимися двумя главными принципами ООП (наследованием и полиморфизмом), которые позволяют создавать семейства связанных типов классов. Вы узнаете о роли виртуальных и абстрактных методов (и абстрактных базовых классов), а также о природе полиморфных интерфейсов. Наконец, в главе будет объясняться роль главного базового класса платформы .NET — `System.Object`.

Глава 7. Структурированная обработка исключений

В этой главе обсуждаются способы обработки аномалий, возникающих во время выполнения, в кодовой базе за счет использования структурированной обработки исключений. Вы узнаете не только о ключевых словах C#, которые дают возможность решать такие задачи (`try`, `catch`, `throw`, `when` и `finally`), но и о разнице между исключениями уровня приложения и уровня системы. Вдобавок в главе будут исследоваться разнообразные инструменты внутри Visual Studio, которые позволяют отлаживать исключения, оставшиеся без внимания.

Глава 8. Работа с интерфейсами

Материал этой главы опирается на ваше понимание объектно-ориентированной разработки и посвящен программированию на основе интерфейсов. Здесь вы узнаете, каким образом определять классы и структуры, поддерживающие несколько линий поведения, обнаруживать эти линии поведения во время выполнения и выборочно скрывать какие-то из них с применением явной реализации интерфейсов. В дополнение к созданию специальных интерфейсов вы научитесь реализовывать стандартные интерфейсы, доступные внутри платформы .NET, и использовать их для построения объектов, которые могут сортироваться, копироваться, перечисляться и сравниваться.

Часть IV. Дополнительные конструкции программирования на C#

В этой части книги вы углубите знания языка C# за счет исследования нескольких более сложных (и важных) концепций. Здесь вы завершите ознакомление с системой типов .NET, изучив интерфейсы и делегаты. Вы также узнаете о роли обобщений, кратко взглянете на язык LINQ (Language Integrated Query) и освоите ряд более сложных средств C# (например, методы расширения, частичные методы и манипулирование указателями).

Глава 9. Коллекции и обобщения

Эта глава посвящена обобщениям. Вы увидите, что обобщенное программирование предлагает способ создания типов и членов типов, которые содержат заполнители, указываемые вызывающим кодом. По существу обобщения значительно улучшают производительность приложений и безопасность в отношении типов. Здесь не только описаны разнообразные обобщенные типы из пространства имен `System.Collections.Generic`, но также показано, каким образом строить собственные обобщенные методы и типы (с ограничениями и без).

Глава 10. Делегаты, события и лямбда-выражения

Целью этой главы является прояснение типа делегата. Выражаясь просто, делегат .NET представляет собой объект, который указывает на определенные методы в приложении. С помощью делегатов можно создавать системы, которые позволяют многочисленным объектам участвовать в двухстороннем взаимодействии. После исследования способов применения делегатов .NET вы ознакомитесь с ключевым словом `event` языка C#, которое упрощает манипулирование низкоуровневыми делегатами в коде. В завершение вы узнаете о роли лямбда-операции C# (=>), а также о связи между делегатами, анонимными методами и лямбда-выражениями.

Глава 11. Расширенные средства языка C#

В этой части главе вы сможете углубить понимание языка C# за счет исследования нескольких расширенных приемов программирования. Здесь вы узнаете, как перегружать операции и создавать специальные процедуры преобразования (явного и неявного) для типов. Вы также научитесь строить и взаимодействовать с индексаторами типов и работать с расширяющими методами, анонимными типами, частичными методами и указателями C#, используя контекст небезопасного кода.

Глава 12. LINQ to Objects

В этой главе начинается исследование языка интегрированных запросов (LINQ). Язык LINQ дает возможность строить строго типизированные выражения запросов, которые могут применяться к многочисленным целевым объектам LINQ для манипулирования данными в самом широком смысле этого слова. Здесь вы изучите API-интерфейс LINQ to Objects, который позволяет применять выражения LINQ к контейнерам данных (например, массивам, коллекциям и специальным типам). Эта информация будет полезна позже при рассмотрении других API-интерфейсов, таких как LINQ to XML, LINQ to DataSet, PLINQ и LINQ to Entities.

Глава 13. Время существования объектов

В финальной главе этой части исследуется управление памятью средой CLR с использованием сборщика мусора .NET. Вы узнаете о роли корневых элементов приложения, поколений объектов и типа `System.GC`. После представления основ будут рассматриваться темы освобождаемых объектов (реализующих интерфейс `IDisposable`) и процесса финализации (с применением метода `System.Object.Finalize()`). В главе также описан класс `Lazy<T>`, позволяющий определять данные, которые не будут размещаться вплоть до поступления запроса со стороны вызывающего кода. Вы увидите, что эта возможность очень полезна, когда нежелательно загромождать кучу объектами, которые в действительности программе не нужны.

Часть V. Программирование с использованием сборок .NET

Эта часть книги посвящена деталям формата сборок .NET. Здесь вы узнаете не только о том, как развертывать и конфигурировать библиотеки кода .NET, но также о внутреннем устройстве двоичного образа .NET. Будет описана роль атрибутов .NET и распознавания информации о типе во время выполнения. Кроме того, объясняется роль исполняющей среды динамического языка (Dynamic Language Runtime — DLR) и ключевого слова `dynamic` языка C#. Наконец, рассматриваются более сложные темы, касающиеся сборок, такие как домены приложений, синтаксис языка CIL и построение сборок в памяти.

Глава 14. Построение и конфигурирование библиотек классов

На самом высоком уровне термин "сборка" применяется для описания двоичного файла `*.dll` или `*.exe`, созданного с помощью компилятора .NET. Однако в действительности понятие сборки намного шире. Здесь будет показано, чем отличаются однофайловые и многофайловые сборки, как создавать и развертывать сборки обеих разновидностей, а также, каким образом конфигурировать закрытые и разделяемые сборки с помощью XML-файлов `*.config` и специальных сборок политик издателя. В ходе этого раскрывается внутренняя структура глобального кеша сборок (Global Assembly Cache — GAC).

Глава 15. Рефлексия типов, позднее связывание и программирование на основе атрибутов

В этой главе продолжается исследование сборок .NET. Здесь будет показано, как обнаруживать типы во время выполнения с использованием пространства имен `System.Reflection`. Посредством типов из этого пространства имен можно строить приложения, способные считывать метаданные сборки на лету. Вы также узнаете, как загружать и создавать типы динамически во время выполнения с применением позднего связывания. Напоследок в главе обсуждается роль атрибутов .NET (стандартных и специальных). Для закрепления материала в главе демонстрируется конструирование расширяемого приложения Windows Forms.

Глава 16. Динамические типы и среда DLR

В версии .NET 4.0 появился новый аспект исполняющей среды .NET, который называется исполняющей средой динамического языка (DLR). Используя DLR и ключевое слово `dynamic` языка C#, можно определять данные, которые в действительности не будут распознаваться вплоть до времени выполнения. Такие средства существенно упрощают решение ряда сложных задач программирования для .NET. В этой главе вы ознакомитесь со сценариями применения динамических данных, включая использование API-интерфейсов рефлексии .NET и взаимодействие с унаследованными библиотеками COM с минимальными усилиями.

Глава 17. Процессы, домены приложений и объектные контексты

Базируясь на хорошем понимании вами сборок, в этой главе подробно раскрывается внутреннее устройство загруженной исполняемой сборки .NET. Целью главы является иллюстрация отношений между процессами, доменами приложений и контекстными границами. Упомянутые темы формируют основу для главы 19, где будет исследоваться конструирование многопоточных приложений.

Глава 18. Язык CIL и роль динамических сборок

Последняя глава в этой части преследует двойную цель. В первой половине главы рассматривается синтаксис и семантика языка CIL, а во второй — роль пространства имен `System.Reflection.Emit`. Типы из этого пространства имен можно применять для построения программного обеспечения, которое способно генерировать сборки .NET в памяти во время выполнения. Формально сборки, которые определяются и выполняются в памяти, носят название динамических сборок.

Часть VI. Введение в библиотеки базовых классов .NET

К этому моменту вы уже должны хорошо ориентироваться в языке C# и в подробностях формата сборок .NET. В данной части книги ваши знания расширяются исследованием нескольких часто используемых служб, которые можно обнаружить внутри библиотек базовых классов .NET, включая создание многопоточных приложений, файловый ввод-вывод и доступ к базам данных посредством ADO.NET. Здесь также раскрывается процесс создания распределенных приложений с применением Windows Communication Foundation (WCF) и API-интерфейс LINQ to XML.

Глава 19. Многопоточное, параллельное и асинхронное программирование

Эта глава посвящена построению многопоточных приложений. В ней демонстрируются приемы, которые можно использовать для написания кода, безопасного к потокам. Глава начинается с краткого напоминания о том, что собой представляет тип делегата .NET, и объяснения внутренней поддержки делегата для асинхронного вызова методов. Затем рассматриваются типы из пространства имен `System.Threading`, а также библиотека параллельных задач (Task Parallel Library — TPL). С применением TPL разработчики .NET могут строить приложения, которые распределяют рабочую нагрузку по всем доступным процессорам в исключительно простой манере. В главе также раскрыта роль API-интерфейса PINQ (Parallel LINQ), который предлагает способ создания запросов LINQ, масштабируемых среди множества процессорных ядер. В завершение главы исследуются разнообразные ключевые слова языка C#, позволяющие интегрировать асинхронные вызовы методов непосредственно в язык.

Глава 20. Файловый ввод-вывод и сериализация объектов

Пространство имен `System.IO` позволяет взаимодействовать со структурой файлов и каталогов машины. В этой главе вы узнаете, как программно создавать (и удалять) систему каталогов. Вы также научитесь перемещать данные между различными потоками (например, файловыми, строковыми и находящимися в памяти). Кроме того, в главе рассматриваются службы сериализации объектов платформы .NET. Сериализация позволяет сохранить состояние объекта (или набора связанных объектов) в потоке для последующего использования. Десериализация представляет собой процесс извлечения объекта из потока в память с целью потребления внутри приложения. После описания основ вы освоите настройку процесса сериализации с применением интерфейса `ISerializable` и набора атрибутов .NET.

Глава 21. ADO.NET, часть I: подключенный уровень

В этой первой из трех глав, посвященных базам данных, представлено введение в API-интерфейс доступа к базам данных платформы .NET, который называется ADO.NET. В частности, здесь рассматривается роль поставщиков данных .NET и взаимодействие с реляционной базой данных с использованием подключенного уровня ADO.NET, который представлен объектами подключения, объектами команд, объектами транзакций и объек-

тами чтения данных. В главе также демонстрируется процесс создания специальной базы данных и первой версии специальной библиотеки доступа к данным (AutoLotDAL.dll).

Глава 22. ADO.NET, часть II: автономный уровень

В этой главе изучение взаимодействия с базами данных продолжается описанием автономного уровня ADO.NET. Вы узнаете о роли типа DataSet и объектов адаптеров данных. Вдобавок вы ознакомитесь с многочисленными инструментами Visual Studio, которые могут значительно упростить создание приложений, управляемых данными. Наряду с этим вы научитесь привязывать объекты DataTable к элементам пользовательского интерфейса и применять запросы LINQ к находящимся в памяти объектам DataSet, используя LINQ to DataSet.

Глава 23. ADO.NET, часть III: Entity Framework

В этой главе исследование ADO.NET завершается рассмотрением роли инфраструктуры Entity Framework (EF). По существу EF предлагает способ написания кода доступа к данным с применением строго типизированных классов, которые напрямую отображаются на бизнес-модель. Здесь вы узнаете о роли класса DbContext инфраструктуры EF, об использовании аннотаций данных для формирования базы данных и о реализации хранилищ для инкапсуляции общего кода, поддержки транзакций, перемещений, параллелизма и перехвата. Кроме того, вы научитесь взаимодействовать с реляционными базами данных с помощью LINQ to Entities. В главе также будет создана финальная версия специальной библиотеки доступа к данным (AutoLotDAL.dll), которая будет применяться в нескольких оставшихся главах книги.

Глава 24. Введение в LINQ to XML

В главе 12 была представлена основная модель программирования LINQ, а именно — LINQ to Objects. Здесь вы углубите свои знания языка LINQ, освоив применение запросов LINQ к документам XML. Первым делом будут описаны сложности с манипулированием данными XML, которые существовали в .NET изначально, когда использовались типы из сборки System.Xml.dll. Затем вы узнаете, как создавать документы XML в памяти, сохранять их на жестком диске и перемещаться по их содержимому с помощью модели программирования LINQ (LINQ to XML).

Глава 25. Введение в Windows Communication Foundation

До этого места в книге все примеры приложений запускались на единственном компьютере. В настоящей главе вы ознакомитесь с API-интерфейсом Windows Communication Foundation (WCF), который позволяет создавать распределенные приложения в симметричной манере независимо от лежащих в их основе низкоуровневых деталей. Здесь будет объясняться конструкция служб, хостов и клиентов WCF. Вы увидите, что службы WCF являются чрезвычайно гибкими, поскольку предоставляют клиентам и хостам возможность использования конфигурационных файлов на основе XML, в которых декларативно указываются адреса, привязки и контракты.

Часть VII. Windows Presentation Foundation

Первоначальный API-интерфейс для построения графических пользовательских интерфейсов настольных приложений, поддерживаемый платформой .NET, назывался Windows Forms. Хотя он по-прежнему полностью доступен, в .NET 3.0 программистам был предложен замечательный API-интерфейс под названием Windows Presentation Foundation (WPF), который быстро стал заменой модели программирования настольных приложений Windows Forms. По существу WPF позволяет строить настольные приложения с векторной графикой, интерактивной анимацией и операциями привязки данных,

используя декларативную грамматику разметки XAML. Более того, архитектура элементов управления WPF предлагает легкий способ радикального изменения внешнего вида и поведения типового элемента управления с помощью правильно оформленной разметки XAML.

Глава 26. Введение в Windows Presentation Foundation и XAML

Инфраструктура WPF позволяет создавать исключительно интерактивные и многофункциональные пользовательские интерфейсы для настольных приложений (и косвенно для веб-приложений). В отличие от Windows Forms инфраструктура WPF интегрирует набор ключевых служб (например, двухмерную и трехмерную графику, анимацию и форматированные документы) в единую унифицированную объектную модель. В этой главе предлагается введение в WPF и расширяемый язык разметки приложений (Extendable Application Markup Language — XAML). Здесь вы узнаете, как создавать приложения WPF без XAML, только с XAML и с комбинацией обоих подходов. В завершение главы рассматривается пример построения специального редактора XAML, который будет использоваться в остальных главах, посвященных WPF.

Глава 27. Программирование с использованием элементов управления WPF

В этой главе будет показано, как работать с элементами управления и диспетчерами компоновки, предлагаемыми WPF. Вы узнаете, каким образом создавать системы меню, окна с разделителями, панели инструментов и строки состояния. Также в главе рассматриваются API-интерфейсы (и связанные с ними элементы управления), входящие в состав WPF, в том числе Documents API, Ink API, команды, маршрутизируемые события, модель привязки данных и свойства зависимости.

Глава 28. Службы визуализации графики WPF

Инфраструктура WPF является API-интерфейсом, интенсивно использующим графику, и с учетом этого WPF предоставляет три подхода к визуализации графических данных: фигуры, рисунки и геометрии и визуальные объекты. В настоящей главе вы ознакомитесь с каждым подходом и изучите несколько важных графических примитивов (например, кисти, перья и трансформации). Кроме того, вы узнаете, как встраивать векторные изображения в графику WPF, а также выполнять операции проверки попадания в отношении графических данных.

Глава 29. Ресурсы, анимация, стили и шаблоны

В этой главе освещены важные (и взаимосвязанные) темы, которые позволят углубить знания API-интерфейса WPF. Первым делом вы изучите роль логических ресурсов. Система логических ресурсов (также называемых объектными ресурсами) предлагает способ именования и ссылки на часто используемые объекты внутри приложения WPF. Затем вы узнаете, каким образом определять, выполнять и управлять анимационной последовательностью. Вы увидите, что применение анимации WPF не ограничивается видеоиграми или мультимедиа-приложениями. И, наконец, вы ознакомитесь с ролью стилей WPF. Подобно веб-странице, использующей CSS или механизм тем ASP.NET, приложение WPF может определять общий вид и поведение для набора элементов управления.

Глава 30. Уведомления, команды, проверка достоверности и MVVM

Эта глава начинается с исследования трех основных возможностей инфраструктуры: уведомлений, команд WPF и проверки достоверности. При рассмотрении уведомлений вы узнаете о наблюдаемых моделях и коллекциях, а также о том, как они поддерживают

данные приложения и пользовательский интерфейс в синхронизированном состоянии. Затем вы научитесь создавать специальные команды для инкапсуляции кода. В разделе, посвященном проверке достоверности, вы ознакомитесь с несколькими механизмами проверки достоверности, которые доступны для применения в приложениях WPF. Глава завершается исследованием шаблона "модель-представление-модель представления" (Model View ViewModel — MVVM) и созданием приложения, демонстрирующего шаблон MVVM в действии.

Часть VIII. ASP.NET

Эта часть посвящена построению веб-приложений с использованием API-интерфейса ASP.NET. Инфраструктура ASP.NET предназначена для моделирования процесса создания настольных пользовательских интерфейсов путем наложения управляемой событиями объектно-ориентированной инфраструктуры поверх стандартного запроса/ответа HTTP. В первых трех главах раскрываются основы веб-программирования и Web Forms, а в финальной главе рассматриваются два новейших элемента в ASP.NET: MVC и Web API.

Глава 31. Введение в ASP.NET Web Forms

В этой главе начинается изучение процесса разработки веб-приложений с помощью ASP.NET. Вы увидите, что сценарии серверной стороны теперь заменены настоящими объектно-ориентированными языками программирования (например, C# и VB.NET). Здесь обсуждается конструкция веб-страницы ASP.NET, внутренняя программная модель и другие ключевые аспекты ASP.NET, такие как выбор веб-сервера и применение файлов `Web.config`.

Глава 32. Веб-элементы управления, мастер-страницы и темы ASP.NET

В то время как предыдущая глава была посвящена созданию объектов `Page` из ASP.NET, в этой главе исследуются элементы управления, которые наполняют внутреннее дерево элементов управления. Здесь вы найдете описание основных веб-элементов управления ASP.NET, включая элементы управления проверкой достоверности, элементы управления навигацией по сайту и разнообразные операции привязки данных. Кроме того, рассматривается роль мастер-страниц и механизма тем ASP.NET, который является альтернативой серверной стороны традиционным таблицам стилей.

Глава 33. Управление состоянием в ASP.NET

Эта глава расширяет ваши знания ASP.NET описанием различных способов управления состоянием в .NET. Подобно классическому ASP в ASP.NET довольно легко создавать cookie-наборы, а также переменные уровня приложения и уровня сеанса. Кроме того, ASP.NET предлагает еще один прием управления состоянием: кеш приложения. После исследования нескольких способов поддержки состояния в ASP.NET вы узнаете о роли базового класса `HttpApplication` и научитесь динамически переключать поведение веб-приложения во время выполнения с использованием файла `Web.config`.

Глава 34. ASP.NET MVC и ASP.NET Web API

В этой главе раскрываются две новейшие связанные друг с другом инфраструктуры ASP.NET: MVC и Web API. Инфраструктура ASP.NET MVC основана на шаблоне "модель-представление-контроллер" (Model View Controller); после ее освоения вы построите приложение MVC. Вы узнаете о формировании шаблонов в Visual Studio, маршрутизации, контроллерах, действиях и представлениях. Затем вы создадите REST-службу Web API для обработки операций создания, чтения, обновления и удаления (CRUD) складских данных (с применением библиотеки `AutoLotDAL.dll`), а напоследок обнови-

те приложение MVC для использования новой службы вместо обращения напрямую к `AutoLotDAL.dll`.

Загружаемые приложения

В дополнение к печатным материалам на веб-сайте издательства для загрузки доступны два дополнительных приложения в формате PDF. В них раскрывается несколько дополнительных API-интерфейсов платформы .NET, которые вы можете счесть удобными в повседневной работе. В частности, вы найдете следующие материалы:

- Приложение А. Программирование с помощью Windows Forms
- Приложение Б. Независимая от платформы разработка приложений .NET с помощью Mono

В первом приложении рассматриваются основы API-интерфейса Windows Forms и предоставляются сведения, необходимые для воссоздания ряда настольных графических пользовательских интерфейсов, которые приводились в начальных главах книги (до описания Windows Presentation Foundation). Второе приложение взято из предшествующего издания этой книги; в нем раскрывается роль платформы Mono более подробно, чем это делалось в главах 1 и 2 настоящей книги. Однако имейте в виду, что экранные снимки в приложении Б были получены в более старой IDE-среде MonoDevelop, которая теперь заменена продуктом Xamarin Studio (рассмотренным в главе 2). Тем не менее, основные примеры кода Mono будут работать ожидаемым образом.

Исходный код примеров

Исходный код всех примеров, рассмотренных в книге, доступен для загрузки на веб-сайте издательства.

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-сайт 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