Посвящается Марлен, Ларе, Саре и Скотту,
вдохновляющим меня на все, что я делаю.

Введение

Язык C# продолжает развиваться и изменяться. Вместе с тем меняется также окружающее его сообщество. В настоящее время все больше и больше разработчиков выбирают C# в качестве своего основного профессионального языка программирования. У этих членов сообщества нет предубеждений, распространенных среди тех, кто начал использовать C# после многих лет опыта работы с каким-то другим языком, основанным на C. Даже у разработчиков, годами применяющих C#, современные темпы изменений породили необходимость выработки многих новых привычек. Язык C# пережил особенно возросший темп инновации после того, как компилятор стал проектом с открытым кодом. К оценке средств, предлагаемых для добавления в язык C#, теперь подключено целое сообщество, а не только небольшой круг экспертов по языку. Кроме того, сообщество принимает участие в проектировании новых средств.

Изменения в рекомендуемых архитектурах и развертываниях также меняют языковые идиомы, которые мы используем как разработчики на C#. Построение приложений за счет компоновки микрослужб, распределенных программ и отделения данных от алгоритмов является частью современной разработки приложений. В языке C# начали предприниматься шаги по охвату этих разных идиом.

Я организовал второе издание книги, принимая во внимание изменения, которые произошли как в языке, так и в сообществе C#. Книга не дает исторического экскурса в изменения языка, а взамен предлагает рекомендации о том, как применять текущую версию C#. Советы, которые были удалены из второго издания, не существенны для сегодняшнего языка C# или нынешних приложений. В новых советах раскрываются новые средства языка и инфраструктуры, а также практики, которым сообщество обучилось на построении нескольких версий программных продуктов, используя C#. Читатели предыдущего издания заметят многочисленные ссылки на книгу *Эффективное программирование на C#, 3-е издание* и отсутствие большого числа советов, которые были ранее. В текущих изданиях я реорганизовал обе книги. В целом предлагаемые 50 советов представляют собой набор рекомендаций, которые помогут вам как профессиональному разработчику применять C# более эффективно.

В книге предполагается, что вы используете C# 7, но она не содержит исчерпывающее толкование новых языковых средств. Подобно всем книгам этой серии здесь даются практические рекомендации о том, как применять такие средства для решения задач, с которыми вы, скорее всего, будете сталкиваться ежедневно. Возможности C# 7 раскрываются в ситуациях, когда новые языковые средства привносят новые и улучшенные способы реализации распространенных идиом. Поиск в Интернете может по-прежнему выдавать более ранние решения, имеющие многолетнюю историю. В настоящей книге эти давнишние рекомендации отмечаются специально и приводятся объяснения, почему усовершенствования языка делают возможными лучшие решения.

Многие рекомендации, предложенные в книге, могут быть проверены анализаторами и исправлениями кода Roslyn. Я поддерживаю хранилище этих ресурсов: `https://github.com/BillWagner/MoreEffectiveCSharpAnalyzers`. Если у вас есть какие-то идеи или желание поучаствовать, то пришлите мне запрос.

Кто должен читать эту книгу?

Книга рассчитана на профессиональных разработчиков, использующих C# в качестве основного языка программирования. В ней предполагается, что читатель знаком с синтаксисом и функциональными возможностями C# и, как правило, имеет опыт работы с языком. Книга не содержит обучающих инструкций по языковым средствам. Взамен в ней обсуждается, как интегрировать все возможности текущей версии языка C# в повседневную разработку. В дополнение к знанию функциональных средств C# допускается наличие у читателя некоторого представления об общеязыковой исполняющей среде (Common Language Runtime — CLR) и оперативном (just-in-time — JIT) компиляторе.

О чем эта книга?

В современном мире данные вездесущи. Объектно-ориентированный подход трактует данные и код как часть типа и его обязанностей. Функциональный подход рассматривает методы как данные. Сервисно-ориетированные подходы отделяют данные от кода, который манипулирует ими. Язык C# развивался для включения идиом, общих для всех упомянутых парадигм, что может затруднить принятие проектных решений. В главе 1 обсуждаются доступные варианты и предлагаются руководящие указания о том, когда и для чего выбираются разнообразные языковые идиомы.

Программирование является по существу проектированием API-интерфейсов. Именно так вы указываете пользователям свои предположения относительно применения вашего кода. Кроме того, API-интерфейсы сообщают многое о вашем понимании потребностей и ожиданий других разработчиков. В главе 2 вы научитесь лучшему способу выражения своих намерений, используя богатую палитру языковых средств C#. Вы увидите, как задействовать ленивую оценку, создавать компонуемые интерфейсы, а также избегать путаницы между разнообразными языковыми элементами в открытых интерфейсах.

Асинхронное программирование на основе задач предлагает новые идиомы для составления приложений из асинхронных строительных блоков. Овладение такими средствами означает возможность создания API-интерфейсов для асинхронных операций, которые ясно отражают, каким образом код будет выполняться, и легки в применении. В главе 3 вы узнаете, как использовать языковую поддержку асинхронного программирования на основе задач для выражения способа выполнения кода с помощью множества служб и с применением различных ресурсов.

В главе 4 рассматривается одно специфическое подмножество асинхронного программирования: многопоточное параллельное выполнение. Вы увидите, как PLINQ делает возможным более легкое разбиение сложных алгоритмов на множество ядер и процессоров.

В главе 5 обсуждается использование C# для динамического программирования. C# является строго типизированным, статически типизированным языком. Однако в наши дни растет число программ, содержащих динамическую и статическую типи-

зацию. Язык C# предлагает способы применения в своих интересах идиом динамического программирования без утери преимуществ статической типизации повсюду в программе. В главе 5 вы узнаете, как использовать динамические возможности и каким образом избегать просачивания динамических типов в программы.

Глава 6 завершает книгу советами о том, как влиться в глобальное сообщество C#. Существует много способов принять участие в этом сообществе и содействовать приведению в порядок языка, который вы применяете каждый день.

Соглашения относительно кода

Демонстрация кода в книге по-прежнему требует соблюдения ряда компромиссов между пространством и ясностью. Я старался извлекать только суть из примеров кода, чтобы четко иллюстрировать конкретное намерение. Часто это означает исключение менее важных частей класса или метода. Иногда это предусматривает отсутствие кода восстановления после ошибок. Открытые методы должны проверять допустимость своих параметров и других входных данных, но в большинстве случаев такой код не будет показан ради экономии пространства. По аналогичным соображениям не приводится код, проверяющий результаты вызова методов, и конструкции `try/finally`, которые нередко входят в состав сложных алгоритмов.

Я также обычно считаю, что большинство разработчиков в состоянии найти соответствующее пространство имен, когда в примерах применяется одно из распространенных пространств имен. Вы можете с уверенностью полагать, что каждый пример неявно включает следующие операторы `using`:

```
using System;
using static System.Console;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

Благодарности

Есть много людей, которых я обязан поблагодарить за их вклад в настоящую книгу. Я удостоился чести быть частью удивительного сообщества C# на протяжении ряда лет. Каждый в списке рассылки C# Insiders (внутри или вне Microsoft) делился идеями и обсуждениями, которые сделали книгу лучше.

Я должен выделить нескольких членов сообщества C#, напрямую помогавших мне с идеями и с воплощением этих идей в конкретные рекомендации. Беседы с Джоном Скитом, Дастином Кемпбеллом, Кевином Пилчем, Джаредом Парсонсом, Скоттом Алленом и, что важнее всего, с Мэдсом Торгерсеном стали основой для многих новых идей в текущем издании.

У меня была замечательная команда технических рецензентов. Джейсон Бок, Марк Михаэлис и Эрик Липперт корпели над текстом и примерами, значительно улучшив качество книги, которую вы держите в руках. Их рецензии были доскональными и полными, и это лучшее, на что только можно было надеяться. Кроме того, они добавили рекомендации, которые помогли мне лучше объяснить многие темы.

Команда в Addison-Wesley — просто мечта для совместной работы. Трина Макдональд была фантастическим редактором, постановщиком задач и движущей силой всего того, что удалось сделать. Она в большой степени опиралась на Марка Ренфро и Оливию

Базейджо, да и сам я тоже. Благодаря их вкладу качество завершенной рукописи было обеспечено от корки до корки. Курт Джонсон продолжает делать невероятную работу по маркетингу технического содержимого. Независимо от того, какой формат вы выбрали, Курт имел отношение к его существованию для этой книги.

Для меня снова большая честь быть частью серии Скотта Мейерса. Он просматривает каждую рукопись, выдвигая предложения и замечания по ее улучшению. Скотт невероятно скрупулезен, а его опыт в разработке программного обеспечения, пусть и не на C#, позволяет ему находить любые области, где я объяснил совет недостаточно четко или не до конца обосновал рекомендацию. Его отзывы, как всегда, были неоценимыми при подготовке настоящего издания.

Как всегда, моя семья дала мне возможность завершить рукопись. Моя жена Марлен терпеливо ждала нескончаемые часы, пока я занимался текстом и примерами. Без ее поддержки я никогда бы не завершил эту или любую другую книгу, равно как и не был бы настолько рад окончанию проектов.

Об авторе

Билл Вагнер является одним из выдающихся разработчиков на C# во всем мире и членом комитета по стандартам C# в организации ECMA. Он занимает должность президента в организации Humanitarian Toolbox, в течение 11 лет удостаивался званий Microsoft Regional Director и .NET MVP, а недавно был назначен в консультативный совет .NET Foundation. Билл сотрудничал с компаниями в диапазоне от стартапов до предприятий, которые улучшали процесс разработки программного обеспечения и расширяли свои команды разработчиков. В настоящее время он трудится в составе команды содержимого .NET Core внутри Microsoft. Он создает учебные материалы для разработчиков, заинтересованных в языке C# и платформе .NET Core. Билл получил степень бакалавра по вычислительной технике в Иллинойском университете в Урбане-Шампейне.

Ждем ваших отзывов!

Вы, читатель этой книги, и есть главный ее критик. 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 интересны любые ваши замечания в наш адрес. Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо либо просто посетить наш вебсайт и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится ли вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас. Отправляя письмо или сообщение, не забудьте указать название книги и ее авторов, а также свой обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию новых книг. Наши электронные адреса:

E-mail: `info@dialektika.com`
WWW: `http://www.dialektika.com`

Наши почтовые адреса:

в России: 195027, Санкт-Петербург, Магнитогорская ул., д. 30, ящик 116
в Украине: 03150, Киев, а/я 152