1 Обзор инструментов и навыков для работы с .NET

В первой главе мы познакомимся с инструментами и навыками, которые вы изучите в этой книге, настроим среду разработки, установив программы Visual Studio 2022, Visual Studio Code и JetBrains Rider, а также разберемся с созданием базы данных и проектов для использования в остальных главах. В книге я буду употреблять названия *Visual Studio*, *Code* и *Rider* для обозначения этих трех редакторов кода.

Упоминая в этой книге слово *«современная»* в контексте платформы .NET, я имею в виду версию 8 и ее предшественницу, такую как .NET 6, произошедшую из .NET Core. Слово *«устаревшая»* касается платформ .NET Framework, Mono, Xamarin и .NET Standard. Современная версия .NET консолидирует перечисленные унаследованные платформы и стандарты.

Каждая глава книги охватывает определенные инструменты и навыки, при этом некоторые из глав более детально посвящены конкретным инструментам или аспектам.

В репозитории GitHub, который я создал для этой книги, размещены полные решения для всех описываемых задач. Он доступен по ссылке github.com/markjprice/tools-skills-net8/.

Перейдя на страницу репозитория GitHub, нажмите клавишу . (точка) или в адресной строке измените домен .com на .dev, чтобы открыть облачный веб-редактор GitHub Codespaces, основанный на ПО Visual Studio Code.

Решать задачи из книги можно как в браузерной версии Visual Studio Code, так и в обычном редакторе кода, а также использовать эти инструменты одновременно. Удобно сравнивать свой код с моим решением и при необходимости копировать/вставлять нужные части.

Знакомство с этой книгой и ее содержимым

Прежде чем перейти к обзору содержания, давайте обозначим контекст. Эта книга — одна из трех, которые я написал о платформе .NET 8, и она охватывает почти все ключевые аспекты, необходимые начинающему разработчику для уверенного старта в работе с .NET.

Сопутствующие книги для продолжения обучения

Эта книга — третья и завершающая трилогию, посвященную платформе .NET 8.

1. В первой книге, *C# 12 and .NET 8 — Modern Cross-Platform Development Fundamentals*, рассматриваются основы языка C# и библиотеки .NET, Blazor и ASP.NET Core для разработки веб-приложений. Она предназначена для последовательного чтения, поскольку навыки и знания, полученные в предыдущих главах, дополняются и необходимы для понимания материала последующих глав.
2. Во второй книге[1] рассматриваются более узкие темы, такие как интернационализация, а также популярные пакеты сторонних разработчиков, в том числе Serilog и NodaTime. Вы научитесь разрабатывать нативные сервисы с *AOT-компиляцией* (ahead-of-time) на минимальных API ASP.NET Core и узнаете, как повысить производительность, масштабируемость и надежность посредством кэширования данных, очередей и фоновых сервисов. С технологиями GraphQL, gRPC, SignalR и Azure Functions вы реализуете дополнительные сервисы. Наконец, вы узнаете, как с помощью Blazor и .NET MAUI создавать графические пользовательские интерфейсы для сайтов, классических и мобильных приложений.
3. В третьей книге, которую вы держите в руках, рассматриваются важные инструменты и навыки, которыми должен обладать каждый профессиональный разработчик .NET. Сюда входят паттерны проектирования и архитектуры решений, отладка, анализ памяти, все важные виды тестирования — модульное, интеграционное, производительности, веб- и мобильное, а также темы, важные для тестирования облачных решений на локальном компьютере, такие как контейнеризация, Docker и .NET Aspire. Наконец, мы рассмотрим, как подготовиться к собеседованию на замещение вакансии разработчика .NET.

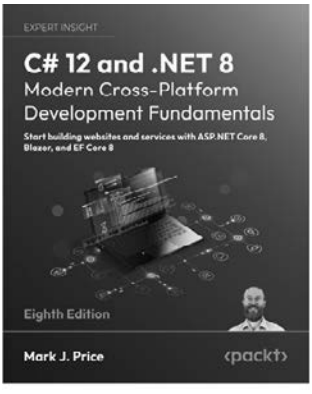
Краткое описание трилогии .NET 8 и ее наиболее важных тем показано на рис. 1.1.

Целевая аудитория

Эта книга рассчитана на следующих читателей.

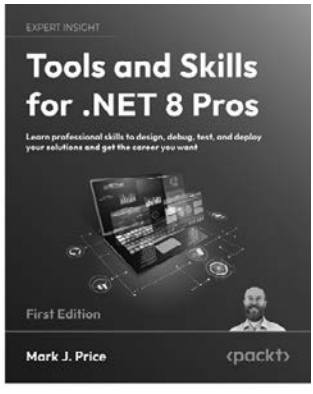
- Читателей моей книги *C# 12 and .NET 8 — Modern Cross-Platform Development Fundamentals*, желающих продолжить обучение.
- Разработчиков с базовыми навыками и знаниями языка C# и платформы .NET, которые стремятся развить практические навыки, освоить распространенные инструменты и повысить уровень профессионализма в разработке на .NET, а также присоединиться к команде разработчиков .NET.

[1] *Прайс М.* .NET 8: приложения и сервисы. Практика создания проектов с использованием Blazor, .NET MAUI, gRPC, GraphQL. 2-е изд. — СПб.: Питер, 2025.



- **Язык C#**, в том числе новые возможности версии C# 12, объектно-ориентированное программирование (ООП), отладка и модульное тестирование.
- **Библиотеки .NET**, в том числе касающиеся работы с числами, текстом, регулярными выражениями, коллекциями, файловым вводом-выводом и данными с помощью EF Core и SQLite.
- **Разработка сайтов и веб-сервисов** с помощью ASP.NET Core и Blazor

- **Дополнительные библиотеки:** интернационализация, многозадачность и пакеты сторонних разработчиков.
- **Дополнительная информация об обработке данных:** SQL Server и Cosmos DB.
- **Дополнительная информация о сервисах:** кэширование, очереди, минимальные API, GraphQL, gRPC, SignalR и Azure Functions.
- **Дополнительная информация о пользовательских интерфейсах:** ASP.NET Core и .NET MAUI



- **Инструменты:** интегрированные среды разработки, отладка, анализ памяти и ИИ-ассистенты.
- **Тестирование:** модульное, интеграционное, производительности, системное и веб, в том числе тесты внедрения зависимостей и инверсий контроля.
- **Проектирование:** паттерны и архитектуры.
- **Развертывание:** непрерывная интеграция/непрерывное развертывание, хостинг в Azure, конвейеры развертывания и автоматизация.
- **Карьера:** подготовка к собеседованию

Рис. 1.1. Трилогия для изучения C# 12 и .NET 8 для начинающих и более опытных читателей

Рассмотрим аналогию.

- Во-первых, начинающий повар может купить книгу, чтобы освоить базовые навыки, ключевые понятия и терминологию, необходимые для приготовления самых популярных блюд.
- Во-вторых, он может взять сборник рецептов, чтобы научиться применять эти знания и навыки на практике — для создания полноценных блюд.
- В-третьих, чтобы стать профессиональным поваром, ему нужно понять, как работает кухня как коллектив, освоить специализированные инструменты и подходы, необходимые для приготовления пищи на большое количество людей, и научиться взаимодействовать с другими поварами в команде.

Именно поэтому я написал три книги о .NET — каждая из них соответствует одному из этих этапов.

Предисловие дает краткий обзор глав. Но так как многие читатели имеют привычку его пропускать, давайте чуть подробнее поговорим о том, почему каждая тема в книге раскрыта так глубоко и последовательно.

Инструменты

Профессиональные разработчики .NET должны быть знакомы со множеством инструментов. Некоторые из них встроены в редакторы кода, например функции отладки и интеграция с системой управления исходным кодом. Другим инструментам требуются отдельные приложения и сервисы, такие как анализ памяти и телеметрия.

Даже начинающие разработчики обычно знают основы работы с инструментами, встроенными в редактор кода: как использовать главное окно редактирования, управлять файлами проекта, устанавливать точки останова, запускать отладку, просматривать код пошагово и запускать проект. Эти темы уже знакомы многим, поэтому в данной книге они не будут детально рассматриваться.

Глава 2 посвящена менее известным инструментам, встроенным в Visual Studio, Visual Studio Code и JetBrains Rider. Основные компоненты — такие как отладчик и средства анализа памяти — будут рассмотрены в отдельных главах. Здесь основное внимание уделяется рефакторингу и настройке редактора кода с учетом стандартов, например `.editorconfig`. Таким образом улучшается совместная работа и обеспечивается единообразие стиля кодирования в команде.

В главе 3 рассматриваются наиболее распространенные задачи, которые .NET-разработчики выполняют с помощью Git при управлении исходным кодом — особенно в командной работе. Git — это распределенная система контроля версий, позволяющая вести разработку в автономном режиме, создавать локальные репозитории, быстро работать с ветками и объединять изменения. Git остается основным инструментом управления версиями в экосистеме .NET. Для него существует множество расширений, интегрируемых с любыми редакторами кода и интерфейсами командной строки. Платформа GitHub, принадлежащая Microsoft, широко используется для хранения репозиториев и совместной работы над проектами.

В главе 4 вы познакомитесь с инструментами отладки, встроенными в редакторы кода. Вы узнаете, как пользоваться отладчиком, размечать код атрибутами для отслеживания событий, а также анализировать потребление памяти. Так вы повысите эффективность и производительность приложений и сервисов.

Глава 5 посвящена внедрению в код инструментов трассировки, сбора метрик и журналирования. Вы узнаете, как применять стандарт OpenTelemetry для сбора телеметрии и мониторинга в реальном времени, что позволяет достичь высокой степени наблюдаемости за состоянием системы.

Навыки

Профессиональному .NET-разработчику важно не только владеть инструментами, но и развивать ключевые навыки — такие как документирование кода, работа с динамическим кодом, а также применение криптографических методов для защиты данных и приложений. В этой книге вы также создадите собственный чат-сервис, использующий модель с поддержкой пользовательских функций.

Глава 6 посвящена эффективным приемам документирования кода, чтобы его было легко поддерживать другим разработчикам. Вы узнаете, как правильно писать комментарии и оформлять документацию для сервисов и API, чтобы ими можно было пользоваться по назначению. Процесс документирования часто помогает выявить участки кода, которые стоит улучшить, — а значит, вы улучшите и архитектуру, и читаемость проекта.

В главе 7 рассматриваются распространенные типы .NET, позволяющие выполнять рефлексию, управлять атрибутами, работать с деревьями выражений и создавать генераторы кода. Эти инструменты особенно полезны при работе в динамически меняющейся среде, когда требуется дополнительная гибкость.

Глава 8 посвящена методам шифрования для защиты данных от несанкционированного доступа, а также хешированию и цифровым подписям — для предотвращения изменений и повреждений данных. Вы также научитесь внедрять проверку подлинности и авторизацию — для защиты приложений от неавторизованного доступа.

В главе 9 вы создадите чат-сервис, интегрированный с большой языковой моделью (LLM), дополненной пользовательскими функциями для обработки и анализа бизнес-данных.

В главе 10 рассматриваются принципы внедрения зависимостей, позволяющие ослабить связность между компонентами, облегчить тестирование, упростить сопровождение и повысить гибкость архитектуры.

Тестирование

Одним из ключевых инструментов .NET-разработчика является тестирование — важнейший компонент обеспечения качества продукта. Тесты охватывают весь жизненный цикл разработки: от проверки отдельных участков кода до сквозного тестирования пользовательского опыта и системной интеграции. Чем раньше в процессе начинается тестирование, тем дешевле и быстрее обходится исправление ошибок.

Глава 11 посвящена методам, помогающим повысить общее качество кода. Модульные тесты легко реализовать неправильно — и тогда они теряют ценность и подры-

вают доверие команды. Но при грамотной реализации они позволяют сэкономить ресурсы и ускорить разработку.

В главе 12 рассматриваются более сложные уровни тестирования, охватывающие все решение целиком, а также методы, направленные на обеспечение безопасности проекта.

Из главы 13 вы узнаете, как использовать библиотеку BenchmarkDotNet для мониторинга и оценки производительности кода. Также рассматриваются нагрузочное и стресс-тестирование с помощью инструментов Bombardier и NBomber, которые помогают прогнозировать потребление ресурсов и оценивать затраты на продакшен-развертывание.

Глава 14 посвящена автоматизированному тестированию API и пользовательских интерфейсов с использованием фреймворка Playwright.

В главе 15 вы познакомитесь с концепцией контейнеризации и узнаете, как применять платформу Docker для виртуализации хостов в сложных архитектурах.

В главе 16 рассматривается .NET Aspire — облачная нативная среда разработки, автоматизирующая локальное развертывание и интеграцию с инструментами, такими как OpenTelemetry для наблюдаемости, Docker для контейнеризации микросервисов, и другие вспомогательные средства.

Построение карьеры в сфере разработки

Заключительная часть книги посвящена теоретическим основам и профессиональному росту. Вы познакомитесь с паттернами проектирования, архитектурными принципами и подходами, а также с рекомендациями по подготовке к собеседованиям на позиции в командах разработчиков и смежных профессиональных ролях.

Глава 17 знакомит с ключевыми паттернами и принципами разработки, включая *SOLID*, а также с другими важными концепциями: *DRY (Don't Repeat Yourself)*, *KISS (Keep It Simple, Stupid)*, *YAGNI (You Ain't Gonna Need It)* и *PoLA (Principle of Least Astonishment)*.

В главе 18 рассматриваются архитектурные подходы к разработке ПО. Вы также изучите инструмент Mermaid, позволяющий создавать архитектурные диаграммы и визуализировать структуру решений.

В главе 19 даны практические рекомендации по поиску работы и подготовке к собеседованиям. Вы узнаете, как составить резюме, произвести впечатление на работодателя и представить свой опыт. Также приведены часто задаваемые вопросы и примеры ответов, которые помогут уверенно пройти собеседование и получить желаемую должность.