Эта книга посвящается моему сыну Арчи

Оглавление

Содержание

Команда DotNetRu приветствует вас, дорогие читатели!

Blazor является инновационной платформой для создания интерактивных веб-приложений с использованием C# и Razor-разметки. Она позволяет разработчикам создавать веб-приложения, минимизируя необходимость в использовании JavaScript. Благодаря Blazor вы сможете писать веб-приложения в том же знакомом и мощном языке, который вы уже знаете и любите, – C#.

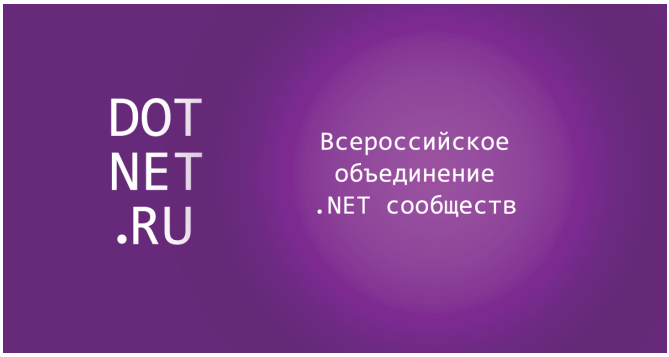
В этой книге вы найдете все, что нужно для глубокого и всестороннего погружения в мир Blazor. Каждая глава аккуратно структурирована, чтобы обеспечить пошаговое и понятное объяснение концепций и технологий Blazor. Независимо от вашего уровня опыта веб-разработки вы найдете полезные советы и практические примеры, которые помогут вам освоить эту технологию.

Эта книга не только предлагает ясные объяснения концепций и шаги для создания приложений Blazor, но и содержит примеры кода и решений к различным реальным сценариям. Вы сможете попробовать работу с Blazor WASM и Blazor Server, узнаете, как создавать пользовательские компоненты, использовать маршрутизацию и многое другое.

Автор создал прекрасную книгу, а мы постарались обеспечить высокое качество перевода, чтобы вы смогли познакомиться с современной терминологией и у вас не возникло сложностей в поиске дополнительной информации в сети Интернет. Мы надеемся, что эта книга поможет вам расширить ваши навыки и вдохновит на новые идеи.

Над переводом работали представители сообщества DotNet.Ru:

- Игорь Лабутин;
- Сергей Бензенко;
- Илья Лазарев;
- Алексей Синютин;
- Евгений Асташев;
- Радмир Тагиров;
- Максим Молоканов;
- Дмитрий Жабин;
- Владимир Майоров;
- Кирилл Вишневский;
- Азамат Сулейманов;
- Андрей Брижак;
- Андрей Беленцов;
- Рустам Сафин;
- Анатолий Кулаков.



Предисловие

Я работаю разработчиком ASP.NET уже более семнадцати лет. Мне нравится работать с ASP.NET Core и C#. Но мне всегда чего-то не хватало…

С юных лет я любил создавать пользовательские интерфейсы для веб-приложений. Когда мне было 15, мы с моим лучшим другом решили создать сайт об играх *Quake*, в которые нам нравилось играть. Он создал серверную часть, а я – пользовательский интерфейс. Помню, как я тратил часы и дни на создание вложенных таблиц и встроенных стилей, чтобы придать сайту нужный нам вид. Сейчас это кажется мучительным процессом, но тогда мне это было по душе. На протяжении всей своей карьеры мне очень нравилось создавать клиентскую часть, но это всегда отвлекало меня от C# и ASP.NET Core. Вместо этого я изучал JavaScript и различные фреймворки и инструменты, популярные в данной экосистеме. Хотя мне нравился JavaScript, в действительности при создании клиентской части веб-приложения я хотел использовать свой любимый язык, C#.

Как-то раз в феврале 2018 г. я наткнулся на видео Стива Сандерсона с конференции NDC в Осло, которая состоялась в 2017 г. (https://youtu.be/MiLAE6HMr10). В своем выступлении он представил проведенный им эксперимент, в котором использовалась переносимая среда выполнения .NET, Dot Net Anywhere, которая была скомпилирована в приложение формата Web Assembly. Стив использовал его в качестве основы для создания фреймворка, позволяющего разрабатывать клиентскую часть веб-приложений с использованием Razor (смесь C#, HTML и CSS), которая полностью работала в браузере. Он назвал его Blazor.

Первая экспериментальная предварительная версия Blazor была выпущена компанией Microsoft 22 марта 2018 г., а новые версии появлялись почти каждый месяц. Я следил за каждой версией, пробовал

новые возможности и писал в блоге о своем опыте. 18 апреля 2019 г. Дэниел Рот опубликовал пост в блоге, в котором объявил, что проект выходит из экспериментальной фазы, и Microsoft обязалась выпустить его в качестве поддерживаемого фреймворка для разработки пользовательских интерфейсов для веб-приложений. Наконец-то, недостающий элемент!

После этого поста в блоге Blazor стал набирать силу. Были добавлены дополнительные модели размещения, позволяющие запускать Blazor в самых разных местах. После выхода .NET 6 мы стали свидетелями одного из самых больших прорывов в этом направлении. Был представлен режим AOT (Ahead-of-time), обеспечивающий значительное повышение производительности приложений Blazor WebAssembly. Эволюция Xamarin, платформа .NET MAUI, позволяет Blazor выйти из браузера и использовать его для создания кроссплатформенных настольных и мобильных приложений.

Эта книга – результат моего путешествия в мире Blazor, начиная с того самого просмотра презентации Стива на конференции NDC в Осло и заканчивая созданием реальных бизнес-приложений. На сегодняшний день я опубликовал более 75 постов о Blazor в своем личном блоге и написал много других публикаций. Blazor привил мне страсть к публичным выступлениям, сначала в группах пользователей .NET и в конечном итоге – на международных конференциях. Мне даже удалось выступить с докладом о Blazor на конференции NDC в Осло в комнате 7, той же комнате, в которой Стив впервые представил свой экспериментальный проект несколькими годами ранее.

Благодарности

Эта книга была одним из самых трудных проектов в моей жизни, и хотя на обложке написано мое имя, ее появление на свет стало возможным только благодаря большому количеству других людей. Пользуясь случаем, хочу сказать им огромное спасибо.

Прежде всего хочу поблагодарить свою жену Робин. Ты была моей опорой на протяжении последних полутора лет. За это время тебе приходилось иметь дело со мной в худшие моменты, но твоя непоколебимая поддержка и ободряющие слова заставили меня поверить, что я могу закончить начатое. Я всегда буду благодарен тебе за это, и я очень тебя люблю.

Также я бы хотел поблагодарить всю мою семью, особенно моего отца, который отговорил меня в какой-то момент все бросить, когда я полностью потерял веру.

Далее я хотел бы поблагодарить замечательных людей из издательства Manning, без которых этой книги не было бы. Брайан Сойер привлек меня, убедил написать коммерческое предложение и убедил, что я справлюсь! Кристен Уоттерсон, мой редактор по развитию почти всего проекта, помогла придать книге тот вид, в котором вы видите ее сегодня. Тони Арритола пришел на выручку в самый последний момент, чтобы помочь довести дело до конца. Эндрю Уэст, технический редактор по разработке, убедился, что мой код не лишен смысла и действительно работает. Наконец, Картикейараджан Раджендран отлично поработал над окончательной технической корректурой. Спасибо также производственной группе Manning за их усердную работу по созданию книги.

Кроме того, я выражаю особую благодарность всем рецензентам за их комментарии и отзывы: Аль Пезевски, Альберто Ачербис, Ашвини Гупта, Бруно Соннино, Грант Колли, Джейсон Хейлз, Джефф Смит, Джим Уилсон, Джон Роудс, Кальян Чанумолу, Марчин Сенк,

Марк Чокли, Майк Тед, Педро Сероменхо, Ричард Майклс, Рохит Шарма, Рон Лиз, Руи Рибейро, Стив Гудман, Таня Уилке, Томас Гет и Уэйн Мазер – вы сделали эту книгу лучше.

Наконец, я бы хотел поблагодарить Стива Сандерсона, Дэниела Рота и всю команду ASP.NET Core из компании Microsoft. Создав Blazor, вы сделали нечто действительно особенное, и это буквально изменило мою жизнь. Теперь я автор книги, выступаю с докладами на международных конференциях и имею действующий статус Microsoft MVP, и все это благодаря Blazor. Спасибо.

О книге

Книга «*Blazor в действии*» была написана с целью превратить вас из новичка в опытного и уверенного разработчика приложений Blazor. Она начинается с описания высокоуровневых концепций, среди которых – модели размещения и компоненты, после чего в ней подробно рассматриваются конкретные функциональные возможности фреймворка, в частности маршрутизация, формы и валидация, а также шаблонные компоненты.

Чтобы помочь вам внедрить различные концепции и возможности, глава за главой мы будем создавать реальное приложение – Blazing Trails. К концу книги у вас будет полноценное эталонное приложение, к которому вы можете обращаться в любое время.

Кому адресована эта книга

Данная книга предназначена для разработчиков, знакомых с основами .NET, C# и веб-технологий (HTML, JavaScript и CSS). Если вы занимаетесь разработкой веб-приложений с использованием Razor Pages или MVC, то погружение в Blazor пройдет для вас гладко. Если вы создавали приложения с использованием веб-API ASP.NET Core и фреймворков JavaScript, в частности React, Vue.js или Angular, то это еще лучше.

Структура книги

Книга состоит из одиннадцати глав и двух приложений:

- глава 1 знакомит вас с Blazor, пользовательскими интерфейсами на основе компонентов и моделями размещения. В ней рассказывается, что такое Blazor и почему можно его использовать, а также почему компоненты лучше подходят для создания пользовательских интерфейсов и как Blazor использует этот

подход. В этой главе также описано, что такое модели размещения, и обсуждаются преимущества и компромиссы каждой из них;

- в главе 2 мы начинаем наше путешествие по созданию приложения Blazing Trails. Она начинается с рассказа о выборе правильного шаблона проекта для нового приложения Blazor, а также способах его создания и запуска. После чего будут рассмотрены ключевые части приложения, а в завершение мы поговорим об организации файлов с разделением по функциональности и о том, как написать свои первые компоненты;
- в главе 3 более подробно рассматривается компонентная модель Blazor. В ней рассказывается, как структурировать компоненты, что такое методы жизненного цикла и в каком порядке они выполняются, как работать с родительскими и дочерними компонентами, а также идет речь о компонентах стилизации и использовании препроцессоров CSS с Blazor;
- в главе 4 рассматривается маршрутизация на стороне клиента, показано, как определять компоненты страницы и перемещаться между ними. В ней также затрагиваются и более сложные темы, в частности передача данных в URL-адресе и навигация программными средствами;
- глава 5 – первая из двух глав, посвященных формам и валидации. В ней рассматриваются такие основы, как использование встроенных компонентов формы Blazor, проверка пользовательского ввода и отправка данных на сервер;
- глава 6 основана на предыдущей главе и охватывает более сложные темы, среди которых – создание собственных компонентов формы, работа с файлами и адаптация формы для редактирования существующих данных;
- глава 7 исследует, как сделать компоненты более пригодными для повторного использования. В ней представлены шаблонные компоненты и способы их дальнейшего улучшения с помощью обобщенных типов;
- в главе 8 показано, как использовать взаимодействие с JavaScript для интеграции существующих библиотек JavaScript в приложение Blazor, а также рассматриваются методы, позволяющие коду C# вызывать код JavaScript, и наоборот;
- глава 9 посвящена защите приложений Blazor. В ней показано, как настроить интеграцию с поставщиком идентификационной информации Auth0;
- в главе 10 рассматривается управление состоянием и реализуется хранилище состояния в памяти. В ней описывается проектирование хранилища состояний и способы хранения состояния с помощью API-интерфейсов локального хранилища браузера;
- глава 11 посвящена тестированию компонентов с использованием фреймворка тестирования bUnit. Рассматриваются пять

ключевых сценариев: тестирование отображаемой разметки, запуск обработчиков событий из тестового кода, имитация аутентификации и авторизации, эмуляция взаимодействия с JavaScript и совместное тестирование нескольких компонентов;
- в приложениях A и B описывается рефакторинг кода, который необходимо выполнять по мере роста приложения. В приложении A описано, как добавить в решение веб-API. Если вы создаете приложение, следуя инструкциям, приведенным в книге, то приложение A должно быть полностью изучено, когда вы будете находиться между главами 4 и 5. В приложении B описывается рефакторинг остальной части приложения для использования веб-API, представленного в приложении A. Приложение B следует изучить после завершения главы 6 и перед началом главы 7.

Соглашения об оформлении программного кода

Эта книга содержит множество примеров исходного кода, как в пронумерованных листингах, так и в виде обычного текста. В обоих случаях исходный код выделен `моноширинным шрифтом, как этот`, чтобы отделить его от остального текста. Иногда используется **`жирный шрифт`**, чтобы выделить код, который изменился по сравнению с предыдущими шагами в главе, например при добавлении новой функции в существующую строку кода.

Во многих случаях исходный код был переформатирован. Мы добавили разделители строк и переделали отступы, чтобы уместить их по ширине книжных страниц. В редких случаях, когда этого оказалось недостаточно, в листинги были добавлены символы продолжения строки (↪). Также из многих листингов, описываемых в тексте, мы убрали комментарии. Некоторые листинги сопровождают аннотации, выделяющие важные понятия.

Исполняемые фрагменты кода можно найти в онлайн-версии книги на странице https://livebook.manning.com/book/blazor-in-action. Исходный код для глав с 2 по 11 доступен в моем репозитории GitHub: https://github.com/chrissainty/blazor-in-action. Код, добавленный в оба приложения, включен в главы, которым они предшествуют.

Весь код из этой книги был создан с использованием .NET 6 SDK и Visual Studio 2022. Однако другие инструменты, в частности Visual Studio Code и .NET CLI или Rider от компании JetBrains, также подойдут для его запуска.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге, – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Manning Publications очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Об авторе

Крис Сэйнти – веб-разработчик с более чем семнадцатилетним стажем. Он работает с Blazor еще со времен первого экспериментального выпуска этого фреймворка в марте 2018 г. и был одним из первых, кто начал вести блог о нем. Крис опубликовал более 75 постов о Blazor в своем блоге, а также пишет статьи для *Visual Studio Magazine*, Progress Telerik и Stack Overflow. Он активный разработчик ПО с открытым исходным кодом и в настоящее время занимается сопровождением некоторых наиболее популярных пакетов NuGet для Blazor, охватывающих интеграцию с API локального хранилища браузера для компонентов пользовательского интерфейса, среди которых – модальные окна и всплывающие сообщения. Когда Крис не сидит за клавиатурой, то выступает на конференциях и рассказывает о Blazor на мероприятиях по всему миру. Все это помогло ему получить награду Microsoft MVP (Most Valuable Professional).

Об иллюстрации на обложке

Иллюстрация на обложке книги «*Blazor в действии*» – *Homme de Oonolaska*, или *Человек из Оноляски* – взята из коллекции Жака Грассе де Сен-Совёра, опубликованной в 1797 г. Каждая иллюстрация в точности отрисована и раскрашена вручную.

В те времена было легко определить, где живут люди и какова их профессия или положение в жизни, просто по одежде. Manning отмечает изобретательность и инициативу бизнеса ИТ книжными обложками, основанными на богатом разнообразии региональной культуры два века назад, которое оживает благодаря иллюстрациям из этой коллекции.

1 Начинаем путешествие в Blazor

В этой главе:

- причины выбрать Blazor для создания приложения;
- преимущества использования компонентов для создания пользовательского интерфейса;
- модели размещения для Blazor.

Сейчас удивительное время для разработчиков на платформе .NET. Мы можем создавать приложения для любой операционной системы, будь то Windows, Linux, iOS, Android или macOS, и, конечно же, разрабатывать потрясающие веб-приложения с помощью ASP.NET MVC, Razor Pages и веб-API – инструментов, которые позволяют создавать надежные и масштабируемые приложения, которые могут работать долгие годы.

Тем не менее этой мозаике давно недоставало одного фрагмента. Общим для всех веб-решений с использованием ASP.NET является то, что они генерируются на сервере. Нам никогда не удавалось использовать мощь C# и .NET для написания клиентских веб-приложений; это всегда было прерогативой JavaScript, но теперь это уже не так.

В первой главе я познакомлю вас с Blazor, революционным фреймворком для разработки клиентских приложений. Будучи созданным на основе веб-стандартов, он позволяет создавать многофункциональные привлекательные пользовательские интерфейсы с использованием C# и .NET. Мы узнаем, как Blazor может сделать процесс

разработки более эффективным и повысить вашу продуктивность, особенно если на вашем сервере также применяется .NET. Мы рассмотрим модели размещения – важную концепцию, которую необходимо усвоить, начиная работу с Blazor. Далее изучим компоненты и преимущества их использования для создания пользовательских интерфейсов. Наконец, мы обсудим причины, по которым следует рассмотреть использование Blazor для вашего следующего проекта.

1.1 Почему стоит выбрать Blazor для создания приложения?

Возможно, самой сложной частью запуска нового проекта в последнее время был выбор технологий – ведь доступно так много вариантов. Это особенно актуально, когда речь идет о разработке клиентской части. Нужно выбрать фреймворк (Angular, React, Vue.js), язык (TypeScript, CoffeeScript, Dart) и инструмент сборки (webpack, Parcel, Browserify). Если команда – новичок в этой экосистеме, то попытка определить, какая комбинация технологий поможет сделать проект успешным, может показаться почти невыполнимой задачей; даже опытным командам будет непросто сделать это!

Рассмотрим основные причины выбора Blazor для создания проекта и то, как он может помочь вам избежать проблем, о которых я только что упомянул:

- *C#, современный и многофункциональный язык* – Blazor создан на основе C#, восьмого по популярности языка, согласно опросу разработчиков Stack Overflow от 2021 г. (http://mng.bz/p240). Это мощный, простой в освоении и универсальный язык программирования. Хотя C# – это объектно ориентированный язык, он перенимает все больше и больше возможностей для реализации функционального подхода, если он вам больше по душе. Статическая типизация помогает разработчикам выявлять ошибки во время сборки, ускоряя жизненный цикл разработки и делая его эффективнее. Данный язык существует уже давно, и в настоящее время вышла одиннадцатая версия C#. Он стабилен, хорошо спроектирован, и у него хорошая поддержка;
- *отличный набор инструментов* – сообществу .NET посчастливилось иметь потрясающие инструменты. Visual Studio – чрезвычайно мощная, многофункциональная и расширяемая IDE (интегрированная среда разработки). Она полностью бесплатна для индивидуальных разработчиков, работы с открытым исходным кодом или для некорпоративных команд до пяти человек. Однако если вам нравится что-то более легковесное, то для этого случая есть Visual Studio Code (VS Code), и это один из самых популярных редакторов кода на сегодняшний день. И Visual Studio, и VS Code доступны для разных платформ. Vi-

sual Studio доступен для ОС Windows и macOS, а VS Code – для Windows, macOS и Linux. Существует также отличная альтернативная IDE от компании JetBrains под названием Rider. Это кросс-платформенное приложение, которое может работать как на Windows, так и на macOS и Linux;

- *полноценная экосистема .NET* – как правило, новые фреймворки, не имеют своего окружения, и должно пройти время, пока вокруг на них будет создана экосистема. Однако в Blazor можно подключиться к уже существующей экосистеме .NET. На момент написания книги приложения Blazor доступны в версии .NET 6 и теоретически могут использовать любые совместимые пакеты NuGet. Я говорю *теоретически*, т. к. некоторые пакеты выполняют действия, которые запрещены в случае с WebAssembly, в частности изменение файловой системы;
- *непредвзятость* – в то время как другие фреймворки определяют способ написания приложения, в Blazor этого нет. Для разработки с Blazor нет предпочтительных паттернов или практик; вы можете писать приложения, используя то, что вам знакомо и удобно. Если вам нравится подход MVVM (модель – представление – модель представления), то пробуйте именно его. Если вы предпочитаете использование стиля как в JavaScript библиотеки Redux, то вперед, выбор за вами;
- *плавная кривая обучения* – если вы уже являетесь разработчиком .NET, то кривая обучения Blazor будет для вас очень плавной. Синтаксис Razor, C#, внедрение зависимостей и структура проекта покажутся вам знакомыми, а поскольку Blazor не требует использования ограниченного набора паттернов, вы можете просто использовать то, с чем вы знакомы и что более эффективно для вас. Все это означает, что вы можете быстрее сосредоточиться на использовании возможностей, а не на изучении фреймворка;
- *повторное использование кода* – если вы используете C# на сервере, то Blazor станет отличным дополнением. Одной из самых неприятных проблем с разными клиентскими и серверными языками является невозможность повторного использования кода. Модели или объекты передачи данных (DTO) должны дублироваться между сервером и клиентом; их нужно постоянно обновлять, синхронизировать. Это можно делать вручную или автоматически с использованием генерирования кода, но это просто еще одна сущность, которую нужно настроить и поддерживать. С Blazor все заточено на C#, и любой общий код можно поместить в общую библиотеку классов .NET и использовать и на сервере, и на клиенте;
- *открытый исходный код* – как и многие проекты Microsoft, Blazor – это проект с полностью открытым исходным кодом, который находится в свободном доступе на GitHub, и вы можете просмотреть, скачать или создать собственную копию. Коман-