Содержание

Предисловие

Если проанализировать самые последние веянья в развитии информационных технологий, можно увидеть, что основным направлением является создание облачных, мобильных и веб-решений, масштабируемых в широких пределах и связанных с управлением большими объемами данных. С появлением этих направлений возникла потребность в инструментах, позволяющих специалистам, таким как вы или я, создавать собственные решения в этой области. Что для этого нужно? Какие архитектуры, инструменты, языки и технологии можно использовать для разработки программ, способных выполняться на самых разных устройствах и легко масштабироваться, и при этом обеспечить высокую надежность решений, простоту их сопровождения, тестирования и многократного использования?

Существует множество инструментов, отвечающих нашим потребностям, но для решения наших задач в полном объеме, их возможностей оказывается недостаточно. Чтобы получить максимальную отдачу, необходим язык, специально предназначенный для преодоления сложностей, возникающих в описанных областях разработки. Он должен иметь встроенные средства для преодоления проблем, связанных с конкуренцией, выполнением асинхронных операций и большими объемами данных, и при этом прозрачно интегрироваться с другими языками, технологиями и инструментами, лучше подходящих для решения других задач. К счастью, такой язык существует и называется F#.

В этой книге я покажу, как использовать язык F# для реализации ключевых элементов облачных, мобильных и веб-приложений, и решения упомянутых проблем. Выразительность, широта возможностей, лаконичность и функциональная природа языка F#, в сочетании с уже известными вам технологиями, такими как ASP.NET MVC, ASP.NET Web API, WCF, Windows Azure, HTML5, CSS3, JavaScript, jQuery и jQuery Mobile, позволят вам создавать удивительные приложения, не только соответствующие, но и превосходящие текущие и будущие требования к ним.

Для кого эта книга

Эта книга предназначена для специалистов с опытом работы в .NET, слышавших о преимуществах F#, имеющих хотя бы общее представление о его синтаксисе, и желающих узнать, как объединять F# с другими технологиями для создания облачных, мобильных и веб-приложений. Если вы совершенно не знакомы с F#, я предлагаю заглянуть в другие книги, описывающие основы программирования на этом языке, такие как книга Криса Смита (Chris Smith) «Programming F#, 3.0» (O'Reilly)[1]. Если вы не знакомы с другими платформами и фреймворками, упоминаемыми в этой книге, такими как ASP.NET MVC, WCF, ASP.NET Web API, Windows Azure, HTML, CSS и/или jQuery Mobile, их описание можно найти во множестве других книг, где вы сможете почерпнуть всю необходимую информацию.

Что необходимо для опробования примеров

Большая часть примеров для этой книги была создана с помощью Visual Studio 2012. Для опробования примеров я рекомендую использовать версию Visual Studio 2012 Professional или выше; однако, большинство примеров будет также работать в среде F# Tools для Visual Studio Express 2012 for Web, анонсированной 12 сентября 2012 в блоге команды разработчиков F#[2]. Загрузить F# Tools для Visual Studio Express 2012 for Web можно по адресу: http://www.microsoft.com/web/gallery/install.aspx?appid=FSharpVWD11. В зависимости от целевой платформы или фреймворка, может потребоваться установить следующие инструменты:

- ASP.NET MVC 4, можно загрузить по адресу: http://www.asp.net/mvc/mvc4.
- Windows Azure SDK и Developer Tools, можно загрузить по адресу: http://www.windowsazure.com/en-us/develop/net/.

Дополнительные библиотеки и инструменты, которые могут потребоваться, упоминаются в соответствующих главах.

---

[1] Крис Смит, «Программирование на F#», ISBN: 978-5-93286-199-8, Символ-Плюс, 2011. – *Прим. перев.*

[2] http://bit.ly/fsharp-blog.

Структура книги

В этой книге рассказывается обо всем, что необходимо знать, чтобы приступить к разработке облачных, мобильных и веб-приложений на языке F#. Кроме того, здесь описывается множество новейших технологий, платформ и библиотек, таких как Windows Azure, jQuery Mobile, SignalR, CouchDB, RavenDB, MongoDB и других. Ниже подробнее описывается, что вы увидите в каждой главе.

Глава 1, «Создание веб-приложений для ASP.NET MVC 4 на языке F#»

В этой главе рассказывается обо всем, что необходимо знать, чтобы приступить к созданию веб-приложений на языке F# с использованием фреймворка ASP.NET MVC 4 на стороне сервера. Здесь также демонстрируются некоторые дополнительные возможности и особенности языка F#, позволяющие писать более элегантный код.

Глава 2, «Создание веб-служб на языке F#»

Эта глава знакомит с инструментами и понятиями, используемыми при создании веб-служб различных типов, включая службы WCF SOAP и HTTP, и особенностями взаимодействий с некоторыми маленькими веб-фреймворками. Здесь также рассказывается об инструментах и приемах модульного тестирования этих веб-служб.

Глава 3, «К облакам! Использование преимуществ Azure»

Эта глава проведет вас через создание веб-приложений и веб-служб на языке F#, выполняющихся под управлением Windows Azure. Дополнительно в ней даются примеры на F# взаимодействий с различными библиотеками Azure. В конце главы будут представлены некоторые замечательные библиотеки и фреймворки на F# для использования на платформе Azure.

Глава 4, «Создание масштабируемых мобильных и веб-приложений»

В этой главе более подробно освещаются вопросы совместного использования F# с другими технологиями для создания масштабируемых решений, позволяющими повторно использовать мобильные и веб-интерфейсы. Глава включает информацию и примеры использования веб-сокетов, библио-

теки SignalR, различных баз данных NoSQL, и многих других механизмов.

Глава 5, «Разработка интерфейсов в функциональном стиле»

Эта глава знакомит с LiveScript, Pit и WebSharper – инструментами, позволяющими, кроме всего прочего, писать клиентский код в функциональном стиле. Они делают возможным создавать комплексные веб-стеки с использованием концепций функционального программирования. Для каждого инструмента перечисляются его преимущества, начальная информация и примеры использования.

В конце книги вы найдете несколько приложений с информацией, которая может вам пригодиться в освоении приемов разработки ультрасовременных облачных, мобильных и веб-приложений, но не относящаяся к темам, обсуждаемым в основных главах.

Приложение A, «Полезные инструменты и библиотеки»

Здесь перечисляются и коротко описываются некоторые инструменты, которые могут облегчить вам жизнь, как разработчика облачных, мобильных и веб-решений.

Приложение B, «Полезные веб-сайты»

Здесь приводятся ссылки на веб-сайты, предлагающие информацию о языке F#, а также инструменты и библиотеки, упоминаемые в книге.

Приложение C, «Клиентские технологии, совместимые с F#»

Здесь дается краткий обзор некоторых технологий, дополняющих F# при разработке мобильных и веб-приложений.

Типографские соглашения

В этой книге приняты следующие соглашения:

Курсив

Курсив применяется для выделения новых терминов, имен файлов и их расширений.

`Моноширинный шрифт`

Применяется для представления листингов программного кода, а также в основном тексте для выделения элементов про-

грамм, таких как имена переменных и функций, базы данных, типы данных, переменные окружения, инструкции и ключевые слова.

Моноширинный жирный

Используется для выделения команд или другого текста, которые должны вводиться пользователем.

Моноширинный наклонный

Обозначает текст, который должен замещаться фактическими значениями, вводимыми пользователем или определяемыми из контекста.

Примечание. Так обозначаются советы, предложения и примечания общего характера.

Внимание. Так обозначаются предупреждения и предостережения.

Использование программного кода примеров

Данная книга призвана оказать вам помощь в решении ваших задач. Вы можете свободно использовать примеры программного кода из этой книги в своих приложениях и в документации. Вам не нужно обращаться в издательство за разрешением, если вы не собираетесь воспроизводить существенные части программного кода. Например, если вы разрабатываете программу и используете в ней несколько отрывков программного кода из книги, вам не нужно обращаться за разрешением. Однако в случае продажи или распространения компакт-дисков с примерами из этой книги вам необходимо получить разрешение от издательства O'Reilly. Если вы отвечаете на вопросы, цитируя данную книгу или примеры из нее, получение разрешения не требуется. Но при включении существенных объемов программного кода примеров из этой книги в вашу документацию, вам необходимо будет получить разрешение издательства.

Мы приветствуем, но не требуем добавлять ссылку на первоисточник при цитировании. Под ссылкой на первоисточник мы подразумеваем указание авторов, издательства и ISBN. Например: «Building Web, Cloud, and Mobile Solutions with F# by Daniel Mohl (O'Reilly). Copyright 2013 Daniel Mohl, 978-1-449-33376-8».

За получением разрешения на использование значительных объемов программного кода примеров из этой книги обращайтесь по адресу permissions@oreilly.com.

Safari® Books Online

Safari Books Online (www.safaribooksonline.com) – это виртуальная библиотека, содержащая авторитетную информацию в виде книг и видеоматериалов, созданных ведущими специалистами в области технологий и бизнеса.

Профессионалы в области технологии, разработчики программного обеспечения, веб-дизайнеры, а также бизнесмены и творческие работники используют Safari Books Online как основной источник информации для проведения исследований, решения проблем, обучения и подготовки к сертификационным испытаниям.

Библиотека Safari Books Online предлагает широкий выбор продуктов и тарифов для организаций, правительственных учреждений и физических лиц. Подписчики имеют доступ к поисковой базе данных, содержащей информацию о тысячах книг, видеоматериалов и рукописей от таких издателей, как O'Reilly Media, Prentice Hall Professional, Addison-Wesly Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology и десятков других. За подробной информацией о Safari Books Online обращайтесь по адресу: http://www.safaribooksonline.com/.

Как с нами связаться

С вопросами и предложениями, касающимися этой книги, обращайтесь в издательство:

O'Reilly Media
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (в Соединенных Штатах Америки или в Канаде)
707-829-0515 (международный)
707-829-0104 (факс)

Список опечаток, файлы с примерами и другую дополнительную информацию вы найдете на сайте книги: http://oreil.ly/building-web.

Свои пожелания и вопросы технического характера отправляйте по адресу: bookquestions@oreilly.com.

Дополнительную информацию о книгах, обсуждения, Центр ресурсов издательства O'Reilly вы найдете на сайте: http://www.oreilly.com.

Ищите нас на Facebook: http://facebook.com/oreilly.

Следуйте за нами на Твиттере: http://twitter.com/oreillymedia.

Смотрите нас на YouTube: http://www.youtube.com/oreillymedia.

Благодарности

Прежде всего я хотел бы поблагодарить мою супругу Мелиссу (Melissa) и дочь Еву (Eva) за то, что поддерживали меня долгие часы сидения за компьютером, когда я работал над этой книгой.

Я также хочу сказать спасибо всем, кто перечислен ниже, за их труд, поддержку и помощь, и за все их лучшие качества!

- Дон Сайм (Don Syme) и все остальные члены команды разработчиков F#;
- Рейчел Румелиотис (Rachel Roumeliotis);
- Илайджа Манор (Elijah Manor);
- Стивен Свенсен (Stephen Swensen);
- Фахад Сухаиб (Fahad Suhaib);
- Райан Райли (Ryan Riley);
- Стеффен Форкман (Steffen Forkmann);
- Антон Таяновский (Anton Tayanovskyy);
- Адам Гранич (Adam Granicz).

Глава 1. Создание веб-приложений для ASP.NET MVC 4 на языке F#

Любая достаточно развитая технология неотличима от магии.

– Сэр Артур Чарльз Кларк (Arthur Charles Clarke)

Я всегда испытывал благоговение перед волшебством и с раннего детства обожал наблюдать за фокусниками. Повзрослев, я стал читать все книги подряд, какие только мог найти, описывающие секреты фокусов, изумлявших меня в течение стольких лет. Вскоре я поймал себя на мысли, что изучать секреты фокусов мне нравится больше, чем смотреть их.

Как заметил сэр Артур Чарльз Кларк, технологии часто сравнимы с магией. Возможно поэтому я так полюбил технические науки. Язык F# относится к этой категории даже больше, чем другие языки, которые мне приходилось использовать в моей карьере программиста. Особенности этого языка открывают такие широкие возможности, что их с полным основанием можно назвать волшебством. Иногда бывает трудно определить, как лучше применить это волшебство на практике для создания еще более масштабируемых облачных, мобильных и веб-приложений, работающих еще лучше, еще быстрее. Эта книга покажет вам, как использовать все возможности языка F# для решения повседневных задач разработки.

В этой главе мы начнем свое путешествие с исследования возможности интеграции F# с фреймворком ASP.NET MVC 4. Здесь вы узнаете, как создать проект, как вести разработку на F# с использованием фреймворка ASP.NET MVC и как применять некоторые дополнительные возможности языка F# для улучшения программного кода. Мы также рассмотрим некоторые темы и приемы, не имею-

щие прямого отношения к ASP.NET MVC 4, но часто используемые вместе с этим фреймворком. На протяжении всей главы мы будем снимать покров тайны с особенностей F#, которые на первый взгляд могут показаться магическими.

В последующих главах мы будем знакомиться с другими платформами, технологиями, библиотеками и механизмами, которые можно использовать в программах на языке F# для создания ультрасовременных облачных, мобильных и веб-решений.

Шаблоны проектов F# ASP.NET MVC 4

О выходе предварительной версии ASP.NET MVC 4 для разработчиков было объявлено после конференции Build Conference во второй половине 2011 года. В феврале 2012 было объявлено о выходе бета-версии ASP.NET MVC 4 и в конце мая 2012 последовала предвыпускная (release candidate) версия. Версия 4 принесла множество улучшений и усовершенствований в и без того полнофункциональный фреймворк ASP.NET MVC. Дополнительную информацию о ASP.NET MVC 4 можно найти на веб-сайте проекта http://www.asp.net/mvc/mvc4.

Самый эффективный способ интеграции F# с фреймворком ASP.NET MVC 4 – использовать преимущества разделения задач, присущие шаблону проектирования «модель–представление–контроллер» (Model–View–Controller, MVC). Это разграничение можно с успехом использовать для усиления экосистемы C# возможностями языка F#. В случае с фреймворком ASP.NET MVC, это достигается путем создания проекта C# ASP.NET MVC, где будут сосредоточены представления и все программные компоненты, выполняющиеся на стороне клиента, и проекта на F#, для реализации моделей, контроллеров и других компонентов, выполняющихся на стороне сервера. На рис. 1.1 показана реализация типичного шаблона проектирования MVC в ASP.NET MVC с обозначением типов компонентов.

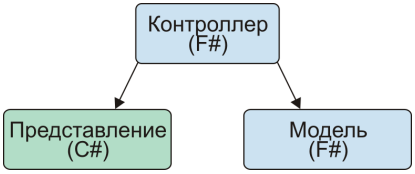

Рис. 1.1. Шаблон проектирования MVC с обозначением типов компонентов

Конечно, приложение с подобной структурой можно создать и вручную, но такой подход быстро становится утомительным. Кроме того, рутинные подготовительные операции являются дополнительным барьером на пути к использованию F# в приложениях на основе ASP.NET MVC. Чтобы помочь устранить эти проблемы, был создан шаблон проекта, доступный в галерее шаблонов проектов Visual Studio Gallery.

Примечание. Для тех, кто по каким-то причинам не может использовать шаблоны проектов ASP.NET MVC 4, в галерее также присутствуют шаблоны ASP.NET MVC 3 и ASP.NET MVC 2. Список большинства доступных шаблонов можно найти по адресу: http://bit.ly/allfsprojecttemplates.

Поиск и установка шаблонов

Благодаря галерее шаблонов проектов Visual Studio Gallery, поиск и установка шаблонов проектов F# ASP.NET MVC 4 выполняются проще некуда. Просто запустите мастер создания нового проекта любым способом, по вашему выбору, – я предпочитаю комбинацию клавиш **Ctrl+Shift+N** – Выберите пункт **Online (В Интернете)** в левой панели, введите текст «fsharp mvc4» в строке поиска в правом верхнем углу окна, выберите шаблон «F# C# MVC 4» и щелкните на кнопке **OK**. На рис. 1.2 изображено окно мастера создания нового проекта в момент, непосредственно перед щелчком на кнопке **OK**.

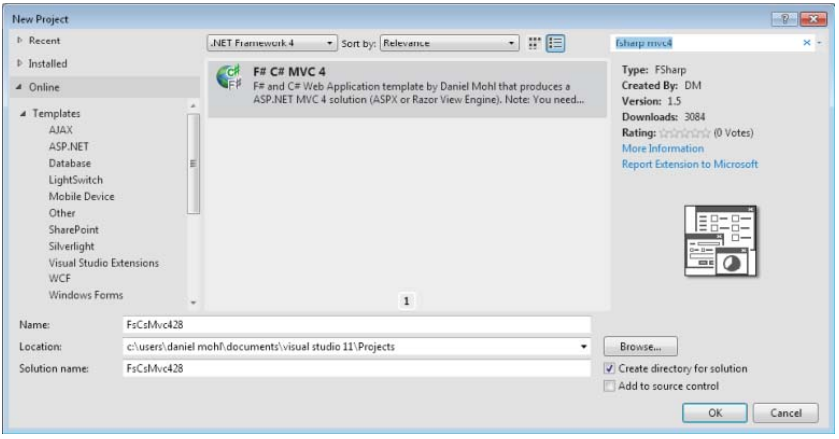

Рис. 1.2. Поиск шаблона проекта в галерее Visual Studio Gallery

Примечание. Описанный порядок действий можно использовать каждый раз при создании нового проекта F# ASP.NET MVC 4, но в действительности достаточно выполнить их один раз. После первичной установки новый шаблон будет доступен в категории «Installed» («Установленные»), в панели слева. Шаблону будет дано имя «F# and C# Web Application (ASP.NET MVC 4)» и вы сможете выбирать его в категории Visual F#→ASPNET.

После щелчка на кнопке **OK** появится диалог (изображенный на рис. 1.3), где можно выбрать тип приложения и механизм представлений (раскрывающийся список **View Engine**), а также необходимость включения дополнительного проекта, где будут находиться модульные тесты. После выбора нужных параметров щелкните на кнопке **OK**. В результате будут созданы все необходимые проекты и установлены пакеты NuGet. В большинстве оставшихся примеров в этой главе будет предполагаться, что в процессе создания приложения был выбран механизм представлений Razor.

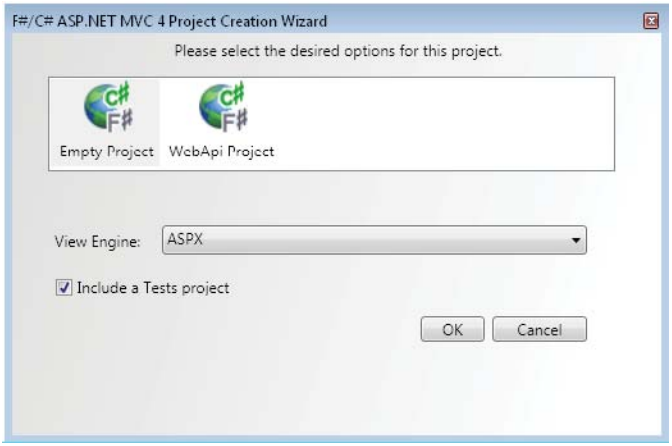

Рис. 1.3. Диалог мастера создания проекта F# ASP.NET MVC

Проект на C#

Если прежде вам приходилось создавать проекты ASP.NET MVC только на C#, приложение C#, созданное выше, покажется вам очень знакомым. В действительности рассматриваемый проект имеет всего три основных отличия:

1. Отсутствует папка *Controllers*.
2. Отсутствует папка *Models*.
3. Файл *Global.asax* не имеет соответствующего ему файла *Global.asax.cs*.

Главная причина этих отличий в том, что перечисленные элементы были перемещены в проект на F#, сгенерированный вместе с данным проектом на C#, но подробнее проект на F# будет рассматриваться в следующем разделе. Файл *Global.asax* не представляет большого интереса. В нем определен лишь один метод для связи с классом на F#. В следующем фрагменте показано содержимое файла *Global.asax*:

```
<%@ Application Inherits="FsWeb.Global" Language="C#" %>
<script Language="C#" RunAt="server">

  // Определение метода Application_Start, вызывающего метод Start
  // класса System.Web.HttpApplication, который наследуется классом Global.
  protected void Application_Start(Object sender, EventArgs e) {
       base.Start();
  }

</script>
```

Проект на F#

Если в диалоге мастера создания проекта (рис. 1.3) был выбран шаблон «Empty Project» (пустой проект), получившийся проект на F# будет очень прост. В проект автоматически будут добавлены все необходимые ссылки на сборки MVC и два файла *.fs*: *Global.fs* и *HomeController.fs*. Я уже коротко упоминал файл *Global.fs* и уверен, что вы уже догадались, что содержит файл *HomeController.fs*. Рассмотрим их подробнее в этом разделе.

Global.fs

Как уже упоминалось, файл *Global.fs* содержит большую часть кода, который обычно находится в файле *Global.asax.cs*, но с некоторыми особенностями, характерными для F#. Первое, что можно в нем заметить, – определение типа `Route`. Это *тип записи* на языке F#, предназначенный для создания определений маршрутов. Типы записей по умолчанию являются неизменяемыми. Поэтому они хорошо согласуются с конкурентной природой Веб, не предполагаю-

щей хранения информации о состоянии. Подробнее о типах записей я буду рассказывать далее в этой книге. Тип Route объявлен, как показано ниже:

```
type Route = { controller : string
               action : string
               id : UrlParameter }
```

Примечание. Тип Route используется только для определения стандартных маршрутов контроллер/действие/ID. Для определения маршрутов других видов необходимо создавать собственные типы.

За объявлением типа Route следует определение класса Global, который наследует класс System.Web.HttpApplication. Код в классе Global выглядит почти так же, как в определении аналогичного ему класса на языке C#, за исключением вызова метода MapRoutes и использования значимых пробелов вместо фигурных скобок для определения области видимости. Главное отличие в вызове метода MapRoutes напрямую связано с типом Route. Для передачи информации о маршрутах методу MapRoutes, благодаря механизму определения типов в языке F#, вместо нового анонимного типа создается новая запись типа Route. Такой синтаксис создания записей называется *выражение записи* (record expression). Ниже приводится определение класса Global, где выделен фрагмент, выполняющий создание записи типа Route:

```
type Global() =
    inherit System.Web.HttpApplication()

    static member RegisterRoutes(routes:RouteCollection) =
        routes.IgnoreRoute("{resource}.axd/{*pathInfo}")
        routes.MapRoute("Default",
                        "{controller}/{action}/{id}",
                        { controller = "Home"; action = "Index"
                          id = UrlParameter.Optional } )

    member this.Start() =
        AreaRegistration.RegisterAllAreas()
        Global.RegisterRoutes(RouteTable.Routes)
```

HomeController.fs

Файл HomeController.fs содержит определение класса HomeController. Он наследует класс Controller и реализует единственное действие с именем Index. Подробнее о контроллерах будет рассказываться ниже в этой главе. Файл HomeController.fs содержит следующий код:

```
namespace FsWeb.Controllers

open System.Web
open System.Web.Mvc

[<HandleError>]
type HomeController() =
     inherit Controller()
     member this.Index () =
          this.View() :> ActionResult
```

Кого-то может смутить комбинация символов :>, выделенная в предыдущем примере. Эта последовательность обозначает приведение типа вверх (upcast) результата вызова this.View() к типу ActionResult. В данном примере приведение к типу ActionResult не является необходимостью, но может потребоваться в некоторых других случаях, поэтому разработчики добавили приведение типа вверх в шаблон с целью демонстрации. Если бы тип возвращаемого значения метода Index был определен явно:

```
member this.Index () : ActionResult = ...
```

тогда приведение типа следовало бы записать так:

```
upcast this.View()
```

Так как в данном конкретном случае приведение типа не требуется, этот метод можно упростить, как показано ниже:

```
member this.Index () =
     this.View()
```

Примечание. Проверка возможности приведения типа вверх (upcast) производится на этапе компиляции, чтобы гарантировать его допусти-