Stylish F#

Crafting Elegant Functional Code for .NET and .NET Core

Kit Eason

Apress®

Stylish F#: Crafting Elegant Functional Code for .NET and .NET Core

Kit Eason
Farnham, Surrey, UK

ISBN-13 (pbk): 978-1-4842-3999-5 ISBN-13 (electronic): 978-1-4842-4000-7
https://doi.org/10.1007/978-1-4842-4000-7

Library of Congress Control Number: 2018963299

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Joan Murray
Development Editor: Laura Berendson
Coordinating Editor: Jill Balzano

Cover image designed by Kit Eason. Contains OS data © Crown copyright and database right 2018

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail rights@apress.com, or visit http://www.apress.com/rights-permissions.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484239995. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

To Val, Matt, Meg, Kate, Noah, and Darwin:
my own persistent collection.

Table of Contents

About the Author

Kit Eason is a software developer and educator with more than 20 years of experience. He has been programming in F# since 2011 and is employed at Perpetuum Ltd., working on an extensive network of energy-harvesting vibration sensors fitted to railway rolling stock. Kit is an avid F# user who is passionate about teaching others. He has contributed to several publications, as well as to the books *Beginning F# 4.0* (Apress 2016) and *F# Deep Dives* (Manning 2014). He often teaches and presents on F#, and his popular videos appear on Lynda.com and YouTube.

About the Technical Reviewer

Quinton Coetzee was born and raised in a small town not too far from Johannesburg, South Africa, which means he can just about remember how to speak Afrikaans. He played paintball at a relatively high level (for a South African team), traveling to the USA and Europe to compete, before moving to the UK in 2011 to pursue career opportunities in London. Since then, he has worked on real-time trading systems and various applications in one way or another related to trading, primarily in F#. As a productivity tool, F# is really hard to beat and would be his first choice in most production environments. That said, he enjoys playing with other functional languages like Clojure and Scala in his spare time.

Acknowledgments

I am grateful for the generous help I received in putting *Stylish F#* together. Thanks to Quinton Coetzee for his exceedingly diligent and constructive technical reviews. To Val Eason for reading every chapter before submission, detecting many typos and poor turns of phrase. To Jon Harrop for providing detailed technical feedback on Chapter 12, and to several other F# community members who reviewed a code sample for Chapter 8. To Jason Heeris for kindly giving permission to reproduce the cartoon in Chapter 1. To Don Syme and the F# community for the never-ending stream of compiler and tooling improvements that propel F# forward. And to Matt Jones and the amazing team at Perpetuum for providing the best working environment I've ever experienced. Thanks also to the tireless crew at Apress: Joan Murray, Jill Balzano, and Laura Berendson.

Any errors, omissions, or plain wrong-headedness are, of course, still my own responsibility.

Introduction

There are three distinct philosophies that you can apply to computer programming. You can think of programming as a *science*, where the measure of progress is how well you discover and reflect fundamental mathematical concepts in your code. You can think of it as a *discipline*, where you seek to establish and follow rules about how code should be written and organized. Or, best of all, you can think of it as a *craft*, where, yes, you apply some of the science and some of the discipline; but you leaven those with a generous helping of human creativity. To do this successfully, you need a fair bit of experience, because crafting something is an inherently intuitive process. This book aims to get you to a level where you can craft code confidently. It does this by distilling and passing on my own experience of writing F# systems in numerous different industries over the past eight years.

Before you start this book, you'll need at least some knowledge of F# syntax and concepts. Maybe you've read some of the wide range of beginner material that's available, and probably you'll have written at least a few simple F# programs yourself. You may well have deeper experience of other languages and environments, such as C# and .NET. That said, I have framed the book so that C# knowledge is not a hard prerequisite: I learned F# before I learned C#, and if I can do it, so can you! Also you definitely don't need any background in Computer Science or functional programming. I don't have even a trace of formal education in either of these areas.

So what's between the covers? In Chapter 1, I'll establish some *principles* that will help us decide whether we are coding well, and say a little bit about why coding stylishly is important. In Chapter 2, we'll pick up the basic tools of our craft and learn to chisel out elegant and reliable *functions*. In Chapter 3, we'll tackle the thorny issue of *missing data*, learning some effective techniques for writing dependable code when certain values might not be available. In Chapter 4, we'll pick up some more powerful crafting tools: the so-called *collection functions*, and explore how you can use them to achieve a surprising amount with very little code. In Chapter 5, we'll delve into the strange world of *immutability*: how you can write programs that achieve a result without explicity changing anything. In Chapter 6, we'll look at *pattern matching*, a concept you may have looked at a little when you learned F# syntax, but which is surprisingly pervasive and

powerful in quality F# code. In Chapter 7, we'll explore *record types*, F#'s go-to structure for storing groups of labeled values. In Chapter 8, we'll cover some ground that might already be familiar to C# developers: *object oriented classes*. In Chapter 9, we'll return to the topic of F# *functions*, and explore what it means for a function to also be a first-class value. In Chapter 10, we'll tame the apparent complexity of *asynchronous and parallel programming*: it needn't be as hard as you think! In Chapter 11, we'll look at *Railway Oriented Programming*, an interesting metaphor you can use to help you think about processing pipelines. In Chapter 12, we'll investigate *performance*: can you really write code that is both elegant and fast? In Chapter 13, we'll establish some useful techniques for *laying out* your code and *naming* items to maximize readability. In Chapter 14, I'll briefly reiterate what we've learned.

As this book is primarily about the language, you'll find relatively few references to other libraries. Of course, to build substantial systems, you'll almost always want to pull in Nuget packages for requirements such as unit testing, serialization, web serving, and so forth. But these libraries constitute a large and fast-changing landscape, so I've chosen to pare things down to the F# essentials for this book. This also means that almost all the examples can be typed in and simply run as F# scripts, and they are provided in script form in the downloadable code samples. In the small number of cases where you need to write a compilable program, I take you through the process alongside the example.

Likewise, you won't find many references to specific integrated development environments (IDEs) such as Visual Studio, Visual Studio Code, Xamarin Studio, or JetBrains Rider. Any of these can be used to edit and run the examples in this book, and all are available as free editions if you don't already have something installed. If your IDE doesn't know about F# "out of the box," simply search online for "F# <your IDE> getting started" to find setup instructions. The samples should work without change on any platform where F# is installed, except you may need to change some paths where the sample code accesses local files.

I very much hope you enjoy sharing my F# experience as much as I enjoyed acquiring it. Don't forget to have fun!