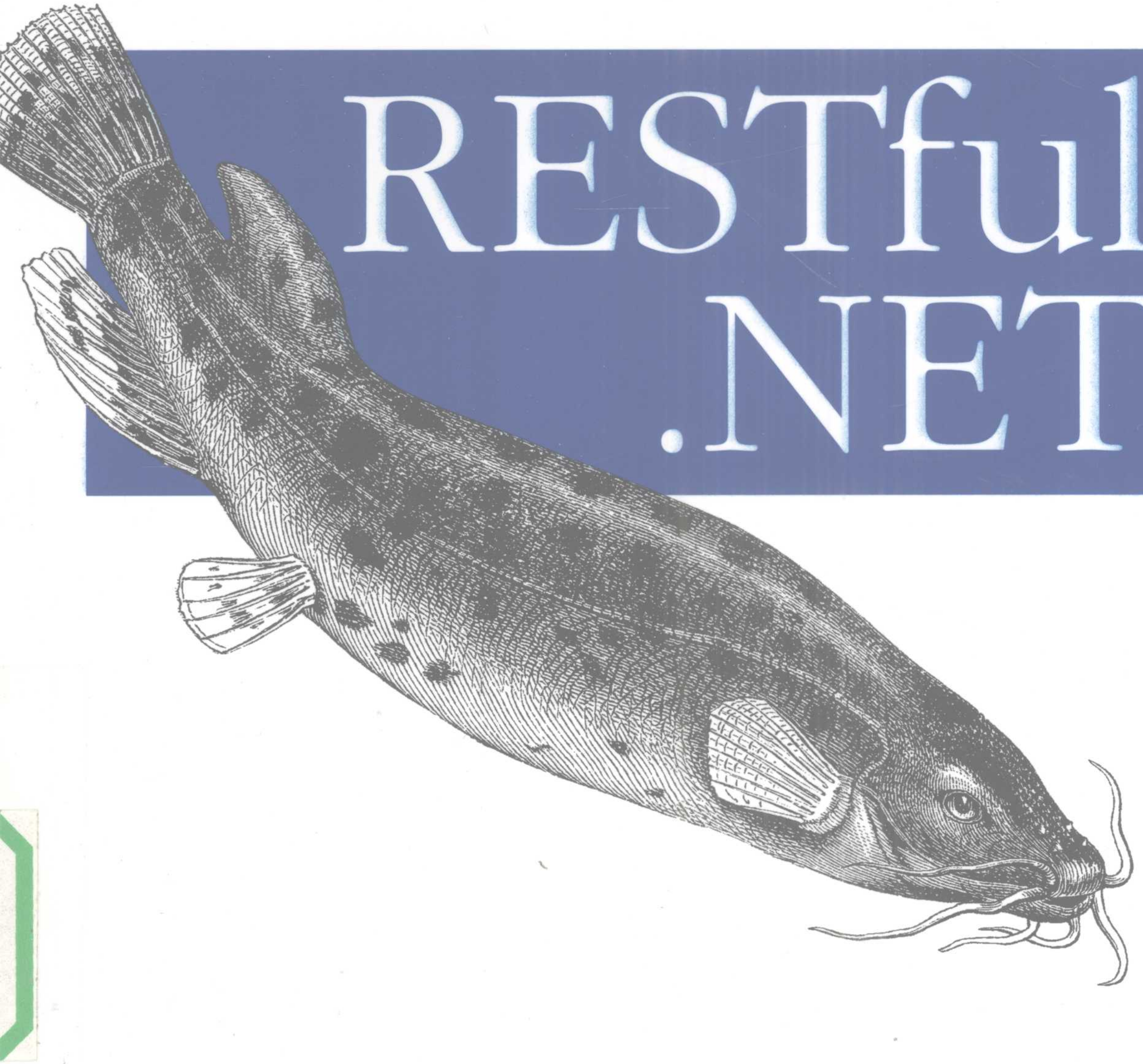
RESTful.NET（影印版）

RESTful .NET



O'REILLY®

東南大學出版社

Jon Flanders 著

RESTful .NET（影印版）

RESTful .NET

Jon Flanders

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc. 授权东南大学出版社出版

东南大学出版社

图书在版编目（CIP）数据

RESTful.NET应用=RESTful. NET：英文/（美）费尔南德斯（Flanders, J.）著.—影印本.—南京：东南大学出版社，2009.5

书名原文：RESTful. NET

ISBN 978-7-5641-1650-7

Ⅰ.R… Ⅱ.费… Ⅲ.主页制作-程序设计-英文

Ⅳ.TP393.092

中国版本图书馆CIP数据核字（2009）第065339号

江苏省版权局著作权合同登记
图字：10-2009-138号

RESTful.NET（影印版）

出版发行：东南大学出版社
地　　址：南京四牌楼2号　　邮编：210096
出 版 人：江　汉
网　　址：http://press.seu.edu.cn
电子邮件：press@seu.edu.cn
印　　刷：扬中市印刷有限公司
开　　本：787毫米×980毫米　16开本
印　　张：19.5印张
字　　数：328千字
版　　次：2009年5月第1版
印　　次：2009年5月第1次印刷
书　　号：ISBN 978-7-5641-1650-7
印　　数：1~2000册
定　　价：43.00元（册）

本社图书若有印装质量问题，请直接与读者服务部联系。电话（传真）：025-83792328

Foreword

I'm an RPC guy in my bones. I spent years of my life working with various remote procedure call technologies, so when SOAP came along, it seemed like the obvious next step on this path. To me, web services meant SOAP, period.

Then REST appeared.

When the RESTful approach first hit the scene, I wrote a short article describing it. At the end, I noted SOAP's broad support, then closed with this:

> Still, even though SOAP is already quite well established, the ideas embodied in REST are worth understanding. Web services are still new, and REST makes a remarkably interesting contribution to the technology.

For a SOAP guy in 2002, I thought I was being quite open-minded. The REST fans didn't see it this way. My inbox sizzled with mail telling me that I was stupid for not immediately seeing REST's innate superiority over the pure evil that was SOAP.

My response was to completely ignore REST for the next several years. I didn't write about it, I didn't speak about it, and I wouldn't even take questions on the topic during talks on web services. I was convinced that REST was the religion of a small band of fanatics, and rude ones at that. The common appellation for a REST fan—RESTafarian—seemed very appropriate to me, derived as it was from the name of an actual religion. These people were true believers, and I couldn't share their faith.

Yet REST was too cool to ignore forever. Once you get your mind around the approach (which doesn't take long—it's simple), REST's beauty is evident. More important, REST's utility is also evident. While SOAP and the WS-* protocols still have a significant rôle, REST is useful in many, many situations. To one degree or another, we're all RESTafarians now.

There's no better evidence of this than Microsoft's embrace of REST in Windows Communication Foundation (WCF). While it's wrong to view this as marking the end of SOAP, WCF's REST support is a big endorsement from what was once the strongest bunch of SOAP advocates. Developers now have a single foundation on which to build all kinds of web services.

But while REST is simple, WCF is not. To really understand and exploit this part of WCF requires a knowledgeable and experienced guide. I don't know anybody who's better suited to this role than Jon Flanders. Along with being one of the smartest people I know, and one of the most capable developers, Jon is first-rate at explaining complicated things.

Even to a long-time RPC guy like me, it's clear that RESTful services will be a big part of the future. This book is the best introduction I've seen to creating and using these services with WCF. If you're a WCF developer looking to enter the RESTful world, this book is for you.

—David Chappell
Chappell & Associates

Preface

I've been working with the Web throughout my entire software engineering career. I started out writing ASP pages and COM components. I then moved into the world of .NET with ASP.NET and ASMX web services.

In 2004, I got involved with BizTalk Server, which pushed me even more into the world of services and XML. I worked with Windows Communication Foundation (WCF) in its early beta stages, before its release in 2007. At that time, the Microsoft world of services was focused on service-oriented architecture (SOA), SOAP, and the WS-* specifications as the preferred methods for building services.

Had I been paying attention, I would have noticed that in 2000 a man named Roy Fielding had written a doctoral dissertation describing the architecture of the Web. By 2000, the Web had arguably become the world's biggest and most scalable distributed application platform. In his dissertation, Fielding examined this platform and distilled from it an architectural style based on the factors that led to its success. He named this architecture REST and suggested it as a way of building not only websites, but also web services.

REST is an architecture that uses the strengths of the Web to build services. It proposes a set of constraints that simplifies development and encourages more scalable designs.

Developers (the majority of whom were outside the Microsoft world) began to adopt this set of architectural constraints shortly after it was proposed (although, to be fair, there were a few inside the Microsoft camp who jumped on the REST technology). Many toolkits embraced REST as the major driver for building applications and services, especially Ruby on Rails, which soared in popularity.

Although WCF isn't tied to SOAP and WS-*, the majority of its programming model was initially geared toward building those kinds of services. The WCF channel model actually did have support for building services using REST, but the WCF programming model lacked explicit support for doing so.

In 2007, a Microsoft program manager named Steve Maine spearheaded an effort to build a REST programming model on top of the WCF infrastructure. This model was released with WCF 3.5 in early 2008.

It was around that time that I read *RESTful Web Services* by Leonard Richardson and Sam Ruby (O'Reilly). After reading and digesting that book, I finally, truly "got it." The "it" that I got wasn't about the technological details, since I understood that part pretty well even before reading the book. The "it" was why people are so enthusiastic about REST. These people are often referred to as RESTafarians, and I now consider myself one of them.

To me, a RESTafarian isn't someone who is religious and argumentative about REST on web forums and blogs (or someone who sends nasty emails to smart people like David Chappell). A RESTafarian is someone who really knows the REST architecture and knows when to apply it in building services. A RESTafarian is someone who understands that using REST's architectural constraints to build services provides a big advantage over RPC-type technology for a large number of systems.

I confess, I am indeed now a RESTafarian. I'll admit it openly and freely. I think using REST should be the first choice when building services, and that RPC should be chosen only if the system requires some particular feature exclusive to RPC technology (like SOAP and WS-*).

After coming to this conclusion, I knew I needed to write this book. I think that all developers deserve to have the tools they need to build highly scalable, loosely coupled services using REST techniques. Hopefully this book will help you learn the ways of REST and how to apply them when developing applications and services using .NET and WCF.

Who This Book Is For

This book is written for .NET developers who are familiar with WCF and REST and who want to learn about using the REST programming model in WCF 3.5.

This book does not teach the fundamentals of WCF. If you aren't familiar with WCF, I highly recommend you read *Learning WCF* by Michele Leroux Bustamante (O'Reilly). Also, while this book does provide some background on REST (in Chapter 1), the book does not focus on the basics of REST. For that, I recommend reading *RESTful Web Services*, followed by Roy Fielding's dissertation, available at *http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm*.

This book is intended to be a companion to both of the books listed above. The samples in this book are all in C#. All of the samples are available on this book's website at *http://www.rest-ful.net/book* in both C# and VB.NET.

How This Book Is Organized

The main chapters of this book are all about WCF 3.5, and the appendixes cover the new features of the WCF 3.5 SP1 upgrade.

Chapter 1, *REST Basics*

Chapter 1 is an introduction to the basic concepts of REST. Again, this book is not intended to be a "learn everything about REST" book. This chapter is a gentle introduction to the concepts of REST. From this chapter you should get the basic ideas of REST, including how resources are identified by unique URIs and how to interact with those resources using the uniform interface of HTTP.

Chapter 2, *WCF RESTful Programming Model*

This chapter introduces the WCF channel and programming models. The purpose of this chapter is to get you oriented in terms of how WCF processes messages and uses those messages to call methods on your services. This chapter should give you a good idea of the plumbing that was added in WCF 3.5 to support this new programming model. It introduces the bindings and hosting infrastructure for building RESTful services as well as the `UriTemplate` class, which is used to map resource URIs onto your methods.

Chapter 3, *Programming Read-Only Services*

`GET` is arguably the most important of the verbs in the HTTP uniform interface. For a high percentage of services, most or all of the functionality is to return read-only data. This chapter will introduce you to the `WebGetAttribute`, which is the mechanism for building resources that return read-only representations.

Chapter 4, *Programming Read/Write Services*

WCF supports the remainder of the uniform interface (`POST`, `PUT`, and `DELETE`) through the `WebInvokeAttribute`. Combined with the `UriTemplate` class, this attribute will enable you to build a complete RESTful service that supports the whole uniform interface.

Chapter 5, *Hosting WCF RESTful Services*

Although this book isn't about WCF in general, one of the key decisions any WCF developer will make is how and where to host services. The RESTful programming model influences that decision, since it is based on HTTP. This chapter will examine special considerations for hosting this type of endpoint.

Chapter 6, *Programming Feeds*

One of the most interesting and exciting features enabled by the RESTful programming model of WCF is the ability to work with feeds. Feeds today are not your father's feeds. Feeds have historically been used (if any technology less than 10 years old can have real history) for publishing web logs (blogs), small technical articles, and the like. Feeds have expanded to include news and other kinds of website data, and are now quickly moving into the Enterprise. Feed readers are built into every modern browser, so they can provide a powerful way to expose corporate data. In this chapter, I'll show you how to build and consume feeds using the WCF feed programming model.

Chapter 7, *Programming Ajax and Silverlight Clients*

Many people see RESTful services as being useful only for exposing data to Ajax-based applications such as mashups, but REST does have reach beyond this type of application. In this chapter, we'll examine WCF 3.5's ability to return data as XML- or JSON-encoded results, as well as the integration between WCF and ASP.NET Ajax.

Chapter 8, *Securing REST Endpoints*

Despite the fact that anti-REST forces often point to a lack of security as a drawback of REST, this is a false argument. RESTful services take advantage of the Web, and the Web has tried-and-true security features.

In this chapter, we'll examine the WCF settings for enabling security and for creating an endpoint that is highly secure.

Chapter 9, *Using Workflow to Deliver REST Services*

Another new piece of functionality in .NET 3.5 is the ability to use Windows Workflow Foundation (WF) workflows to implement and consume services. The RESTful programming model can be used on top of this facility.

This chapter focuses on both stateless and stateful workflow models for implementing RESTful services.

Chapter 10, *Consuming RESTful XML Services Using WCF*

WCF is used as much for building service clients as it is for building services themselves. The same is true of the RESTful programming model. In this chapter, we'll take a RESTful service, SQL Server Data Services (which is a cloud-based storage system), and decompose it into a WCF service contract that can invoke the service through the WCF programming model.

Chapter 11, *Working with HTTP*

Most RESTful implementations use HTTP as the application protocol. In this chapter, I'll talk about how to interact between the WCF programming model and the HTTP request and response messages. Also, we'll look at a couple of slightly more advanced HTTP features and how to use them with your RESTful services in WCF.

Appendix A, *WCF 3.5 SP1*

WCF 3.5 SP1 was released just as this book was being finalized. Appendix A discusses the SP1 improvements and contains a list of new features found in the upgrade, including the new `UriTemplate` syntax and the new support for AtomPub.

Appendix B, *ADO.NET Data Services*

.NET 3.5 SP1 includes ADO.NET Data Services (codename *Astoria*), which provides you with the ability to use a prebuilt WCF service contract to expose a data-backed object model through AtomPub. This appendix shows you how to use ADO.NET Data Services and discusses why you might choose to use it instead of writing your own custom RESTful service endpoints.

Appendix C, *ADO.NET Entity Framework Walkthrough*

ADO.NET Data Services will use any data-backed object model to expose an AtomPub service, but is optimized for use with the ADO.NET Entity Framework (EF). Although EF doesn't have anything to do with RESTful services, I have included this appendix to demonstrate how you can use EF to implement the types of services that are explained in Appendix B.

What You Need to Use This Book

To run the samples provided throughout this book, you need to have Visual Studio 2008 (any version) installed. If you want to work with the code in the appendixes, you will require .NET 3.5 SP1, Visual Studio 2008 SP1, and SQL Server Express or above.

Conventions Used in This Book

The following typographical conventions are used in this book:

Plain text
: Indicates menu titles, menu options, menu buttons, and keyboard accelerators (such as Alt and Ctrl).

Italic
: Indicates new terms, URLs, email addresses, filenames, file extensions, pathnames, directories, and Unix utilities.

`Constant width`
: Indicates commands, options, switches, variables, attributes, keys, functions, types, classes, namespaces, methods, modules, properties, parameters, values, objects, events, event handlers, XML tags, HTML tags, macros, the contents of files, or the output from commands.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values.

This icon signifies a tip, suggestion, or general note.

This icon indicates a warning or caution.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books *does* require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation *does* require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*RESTful .NET* by Jon Flanders. Copyright 2009 Jon Flanders, 978-0-596-51920-9."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

How to Contact the Author

Feel free to look at this book's web page at *http://www.rest-ful.net/book*. You can also email me at *jon.flanders@gmail.com*.

Comments and Questions

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

http://www.oreilly.com/catalog/9780596519209

To comment or ask technical questions about this book, send email to:

bookquestions@oreilly.com

For more information about our books, conferences, Resource Centers, and the O'Reilly Network, see our website at:

http://www.oreilly.com

Acknowledgments

There are a number of people I'd like to thank for helping to make this book possible.

I'd like to thank John Osborn, my editor at O'Reilly, for helping me to write this book and, of course, for putting up with the delays.

Thanks to David Chappell for writing an incredible Foreword. David is one of the smartest people I run into in my travels, and is also one of the nicest and most sincere. Thanks David.

I want to thank all of the technical reviewers: Aaron Lerch, Dare Obasanjo, Aaron Skonnard, Drew Miller, Matt Milner, Michele Bustamante, Julia Lerman, Dominick Baier, Sam Gentile, Dave Chappell, Brian Noyes, Steve Resnick, and Matthew Fowle. If you think part of the book is good, the reviewers deserve credit. If there is any part of the book that you don't like, the fault is exclusively my own.

I also want to thank some people at Microsoft: Steven Maine for pushing to get this programming model into WCF, and Don Box for being the person who helped to get my career started.

The people who deserve the most thanks are members of my family. I want to thank my wife Shannon Ahern for enabling me to do the things I love to do, through her love and support (and also for being an incredible technical editor—if there are any spelling or grammar mistakes in this book, they were introduced after her editing pass). I also need to thank our children: Christian, Raiden, Austin, Parker, and Catherine for putting up with long hours of writing, and having to quiet their normal level of enthusiasm to give me an environment in which to work.

O'Reilly Media, Inc.介绍

O'Reilly Media, Inc.是世界上在 UNIX、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly Media, Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc.是最稳定的计算机图书出版商 —— 每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media, Inc.具有深厚的计算机专业背景，这使得O'Reilly Media, Inc.形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc.所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc.还有许多固定的作者群体 —— 他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media, Inc.依靠他们及时地推出图书。因为O'Reilly Media, Inc.紧密地与计算机业界联系着，所以O'Reilly Media, Inc.知道市场上真正需要什么图书。

出版说明

随着计算机技术的成熟和广泛应用，人类正在步入一个技术迅猛发展的新时期。计算机技术的发展给人们的工业生产、商业活动和日常生活都带来了巨大的影响。然而，计算机领域的技术更新速度之快也是众所周知的，为了帮助国内技术人员在第一时间了解国外最新的技术，东南大学出版社和美国 O'Reilly Meida, Inc.达成协议，将陆续引进该公司的代表前沿技术或者在某专项领域享有盛名的著作，以影印版或者简体中文版的形式呈献给读者。其中，影印版书籍力求与国外图书“同步”出版，并且“原汁原味”展现给读者。

我们真诚地希望，所引进的书籍能对国内相关行业的技术人员、科研机构的研究人员和高校师生的学习和工作有所帮助，对国内计算机技术的发展有所促进。也衷心期望读者提出宝贵的意见和建议。

最新出版的影印版图书，包括：

- 《应用程序性能测试艺术》（影印版）
- 《算法技术手册》（影印版）
- 《学习 OpenCV》（影印版）
- 《Adobe AIR 1.5 Cookbook》（影印版）
- 《构建嵌入式 Linux 系统 第二版》（影印版）
- 《Web 应用程序通用设计》（影印版）
- 《Web 2.0 策划指南》（影印版）
- 《浅入浅出 C#》（影印版）
- 《容量规划的艺术》（影印版）
- 《RESTful.NET》（影印版）
- 《学习 JavaScript 第二版》（影印版）
- 《SQL 技术手册 第三版》（影印版）

Table of Contents