Programming MapPoint in .NET

People use maps all the time for a wide variety of tasks, but getting your own data into maps has been difficult until recently. Microsoft's MapPoint program, MapPoint Web Services, and Virtual Earth offer developers several different ways to integrate their own information with commercial map and demographic data. Adding your own data to that mix is now as simple as telling MapPoint where to find your information and how you want it presented.

*Programming MapPoint in .NET* will show you how to combine MapPoint's capabilities with the strengths of .NET to integrate mapping with your own projects. You'll learn how to:

- Use MapPoint to generate maps that your own application displays
- Create interactive maps using the MapPoint ActiveX control
- Generate driving directions using both MapPoint Web Services and the desktop application
- Connect a GPS device to your own application for navigation with MapPoint
- Integrate your own data with MapPoint's geographic and demographic data
- Use other companies' data services in conjunction with MapPoint's Web Services
- Explore the planet with Visual Earth and add your own information to enrich the users' experience

Whether your projects are intended to help users get from place to place or to present information for analysis visually, *Programming MapPoint in .NET* gives you all the tools you need to make geographic information accessible.

Chandu Thota is Microsoft's point man on MapPoint. You can see his picture at http://msdn.microsoft.com/mappoint/. His weblog at http://blogs.msdn.com/cthota/ is a key resource for MapPoint developers.
Programming MapPoint in .NET

Chandu Thota

© O’Reilly®
Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo
To my wife, Taarinya
# Table of Contents

Foreword ................................................................. ix  
Preface ................................................................. xi  

1. **Hello, MapPoint!** ......................................................... 1  
   Location-Based Application Categories 1  
   Location Application Architectures 2  
   Developing Location-Enabled Applications 4  
   Developing Location-Aware Applications 11  
   How It All Fits Together 13  
   Where Are We? 14  

Part I.  **MapPoint 2004**

2. **Programming with MapPoint 2004** .......................... 17  
   The MapPoint 2004 Object Model 17  
   Programming with MapPoint 2004 API 19  
   Programming the MapPoint ActiveX Control 39  
   Cleaning Up After You’re Done 52  
   Dealing with Latitude and Longitude 53  
   Routing in MapPoint 2004 58  
   Where Are We? 63  

3. **Working with Data in MapPoint 2004** ...................... 64  
   Understanding the MapPoint 2004 Data API 64  
   Understanding Data Maps 66  
   Working with MapPoint Demographic Data 80
Working with Your Business Data 90
Querying a MapPoint DataSet 101
Working with Shapes 105
Working with Territories 114
Where Are We? 117

4. Advanced MapPoint 2004 Programming ................................. 118
   Interfacing MapPoint 2004 with a GPS Device 118
   Integrating Your Applications with MapPoint 2004 133
   Performance Considerations 140
   Where Are We? 141

Part II. MapPoint Web Service

5. Programming MapPoint Web Service .................................. 145
   How Does MapPoint Web Service Work? 146
   Getting Started with MapPoint Web Service 147
   Preparing Your Development Environment 150
   Understanding MapPoint Web Service Object Model 159
   Where Are We? 166

6. MapPoint Web Service Find APIs .................................... 167
   Understanding Find APIs 167
   Understanding Entities and Entity Relationships 170
   Data Sources and Countries/Regions 173
   Working with Find Methods 176
   Asynchronous Programming with Find Service 196
   Optimizing Find Call Performance 205
   Globalizing Find 208
   Where Are We? 209

7. MapPoint Web Service Route APIs ................................... 210
   Understanding Route APIs 210
   Calculating a Route 214
   Displaying Details of a Route 219
   Where Are We? 221
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>MapPoint Web Service Render APIs</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td>Understanding Render APIs</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td>Rendering Maps</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td>Map Interaction</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td>Asynchronous Programming</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td>Where Are We?</td>
<td>254</td>
</tr>
<tr>
<td>Part III</td>
<td>MapPoint Location Server</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Programming Microsoft Location Server</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td>Microsoft Location Server Terminology</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td>How Does Location Server Work?</td>
<td>258</td>
</tr>
<tr>
<td></td>
<td>Microsoft Location Server Architecture</td>
<td>260</td>
</tr>
<tr>
<td></td>
<td>Installing the Location Server</td>
<td>262</td>
</tr>
<tr>
<td></td>
<td>Accessing the Location Service APIs</td>
<td>264</td>
</tr>
<tr>
<td></td>
<td>Accessing the Location Server APIs</td>
<td>264</td>
</tr>
<tr>
<td></td>
<td>Microsoft Location Server Supported Providers</td>
<td>266</td>
</tr>
<tr>
<td></td>
<td>Where Are We?</td>
<td>266</td>
</tr>
<tr>
<td>10</td>
<td>Programming with Location Server APIs</td>
<td>267</td>
</tr>
<tr>
<td></td>
<td>Programming with Location Server Web Service</td>
<td>267</td>
</tr>
<tr>
<td></td>
<td>Programming with the Location Server Management API</td>
<td>277</td>
</tr>
<tr>
<td></td>
<td>Comparing Location Server API to Location Web Service API</td>
<td>287</td>
</tr>
<tr>
<td></td>
<td>Where Are We?</td>
<td>288</td>
</tr>
<tr>
<td>Part IV</td>
<td>MSN Virtual Earth</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Programming with Virtual Earth</td>
<td>291</td>
</tr>
<tr>
<td></td>
<td>Anatomy of Virtual Earth</td>
<td>291</td>
</tr>
<tr>
<td></td>
<td>Programming with MapControl.js</td>
<td>295</td>
</tr>
<tr>
<td></td>
<td>Where Are We?</td>
<td>309</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>A. Managing Your Data on MapPoint’s Customer Services Site</td>
<td>311</td>
<td></td>
</tr>
<tr>
<td>B. Working with Polygons</td>
<td>321</td>
<td></td>
</tr>
<tr>
<td>C. Implementing Spatial Search</td>
<td>326</td>
<td></td>
</tr>
<tr>
<td>Using SQL Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td>333</td>
<td></td>
</tr>
</tbody>
</table>
Foreword

It’s not surprising that discovery has always been rooted in the art and science of mapping. Maps represent many complex relations and hordes of data, yet a child is intuitively able to navigate and comprehend them. Maps have long been the information windows through which explorers and others have peered to find answers. Maps tell the story of the past, record the present, and reveal the future.

Today is no different from the past in that most information—whether business or personal—has a strong relational tie to the place where we reside. The real world provides a commonly understood framework for the endless fields of digital data—whether it lives on the Internet, your corporate network, your desktop, or your mobile device.

The conventional map is changing quickly into a digital representation of the real world. Traditional maps are quickly merging with immersive imagery, local content, 3-D models, and real world sensors. The digital map will have life through its vibrant contributing community of authors and experts that fulfill the dream of global access to local knowledge. Imagine a world where you pivot information in a highly visual place to discover what it’s like somewhere else. Layers of information—whether you seek the news, classifieds, weather, stores, restaurants, photos, local businesses, tourist attractions, sporting events, music venues, or recreational activities—are easily discoverable. Your friends and colleagues share favorites and experiences among each other. The community adds pictures and content, corrects data and information, and provides currency and expertise in specialized knowledge until the map takes on a life of its own. If these concepts excite you, you share the passion of the Virtual Earth and MapPoint team.

For many years, unleashing the power of location technology within business or consumer applications was daunting. Geographic Information Systems were limited to companies with inelastic needs and ample resources. Developers were relinquished to niche departmental, non-mainstream applications due to the heavy lifting involved with data management, securing appropriate development environments, and the lack of an easy way to distribute mapping functionality. Deploying a simple real
estate application to a national field sales team would have required a developer to distribute hundreds of megabytes of aging geographic data with desktop code only to have IT revisit this task frequently to update the application with new features or new street data. Web services provided developers with the first means to build broad, scalable, horizontal applications across the network. Web services abstract the developer from the intricacies of GIS and the anchor of large data sets and provide the developer with a means to update features and data quickly and inexpensively.

As a team set to build platforms for location-based application development, we embraced the change years ago with the first commercial SOAP/XML MapPoint Web Service and continue to lead the next charge with Virtual Earth. Virtual Earth, in conjunction with MapPoint Web Service, is both a platform and a user experience that revolutionizes the way users experience location information both online and offline. Given the simplicity of adding location technology to your application, the awesome user appeal of an immersive, well understood real-world user dimension, and the realization of significant business and user benefits, there is no better time to develop location technology and services.

Chandu Thota is an accomplished developer on the Microsoft Virtual Earth and MapPoint team, as well as an influence on much of the early thinking in the next wave of the Internet and mapping technologies. He knows firsthand the best way to develop applications in a quick and effective way using MapPoint technologies. In this book, he narrates a developer story about a platform that has been his work, play, and passion at Microsoft. Whether you are an enterprise developer or hobbyist programmer, *Programming MapPoint in .NET* will set you apart from other developers in the field.

Unleash your application and data with the power of location,

—Stephen Lawler

*General Manager, Virtual Earth and MapPoint*
Preface

Location is everything! Whether you are trying to find driving directions to the airport or looking for a good restaurant in town, location information has become an essential ingredient of our daily life. Enterprises today consider providing location-based services to their customers an important part of their service offerings. Using MapPoint products and services, you can answer questions such as:

- Where am I?
- How do I get from here to there?
- Where is the nearest coffee shop?

Whether you want to answer these questions using an online application, a web service, or a mobile phone in real-time, or offline, this book is for you. MapPoint provides an integrated set of products, servers, and services that helps enterprises improve their customers’ experience by applying mapping and location.

If you are looking for more programming resources on MapPoint 2004, MP2K Magazine provides excellent online resources and the most up-to-date technical articles on MapPoint 2004 programming. Check it out at http://www.mp2kmag.com.

Who Should Read This Book

Programming MapPoint in .NET will be useful to anyone who wants to develop a location-based application using the following MapPoint technologies:

- MapPoint 2004
- MapPoint Web Service
- Microsoft Location Server
- MSN Virtual Earth

This book provides a jump-start for working with these technologies with in-depth discussions about the core MapPoint technologies provided in C#.
What’s in This Book

This book is organized into 4 major sections with a total of 11 chapters. Each product/technology has a dedicated section in the book:

Chapter 1, Hello, MapPoint!
Introduces the MapPoint suite of products and technologies, setting the stage by discussing the basics of each technology and usage scenarios.

Part I, MapPoint 2004

Chapter 2, Programming with MapPoint 2004
Covers programming with the MapPoint 2004 ActiveX control and MapPoint 2004 object model for rendering maps, finding places and addresses, and calculating routes.

Chapter 3, Working with Data in MapPoint 2004
Covers dealing with business data, rendering data maps, and adding thematic shapes using geographic data.

Chapter 4, Advanced MapPoint 2004 Programming
Covers integration with GPS for obtaining real-time location and extending MapPoint capabilities by writing add-ins.

Future versions of MapPoint (such as MapPoint 2006) are fully backward-compatible with the MapPoint 2004 APIs, so the contents of these chapters are still relevant for MapPoint 2006 and future backward-compatible versions of MapPoint.

Part II, MapPoint Web Service

Chapter 5, Programming MapPoint Web Service
Introduces the basics of programming with MapPoint Web Service.

Chapter 6, MapPoint Web Service Find APIs
Covers creating applications using the Find APIs of the MapPoint Web Service, including techniques for finding places, addresses, and nearby points of interest.

Chapter 7, MapPoint Web Service Route APIs
Covers programming with the Route APIs, such as calculating routes and getting driving directions.

Chapter 8, MapPoint Web Service Render APIs
Covers programming with the Render APIs available with MapPoint Web Service, including rendering maps, routes, LineDrive maps, and polygons.
Part III, *MapPoint Location Server*

Chapter 9, *Programming Microsoft Location Server*
Covers the basics of programming with Microsoft Location Server and deployment scenarios.

Chapter 10, *Programming with Location Server APIs*
Covers programming with the Location Server Web Service to get real-time location using mobile phones; also covers the basics of managing the Location Server, contacts, and privacy settings.

Part IV, *MSN Virtual Earth*

Chapter 11, *Programming with Virtual Earth*
Covers the basics of programming with the new MSN Virtual Earth (undocumented) APIs.

Appendixes

Appendix A, *Managing Your Data on MapPoint’s Customer Services Site*
Provides programming information for the MapPoint Customer Data Service.

Appendix B, *Working with Polygons*
Provides basic information on understanding polygons in MapPoint Web Service.

Appendix C, *Implementing Spatial Search Using SQL Server*
Provides a solution for implementing a proximity search within your enterprise network that doesn’t require you to upload your data to MapPoint Web Service.

This book covers the most common application development scenarios with the MapPoint platform. If you feel that something important has been left out that should be included, let me know. I’ll work to get it in a future edition. For contact information, see the “We’d Like Your Feedback!” section later in the preface.

**Conventions in This Book**

The following typographical conventions are used in this book:

*Italic*
Introduces new terms, URLs, commands, file extensions, filenames, directory or folder names, and UNC pathnames.

*Constant width*
Indicates command-line elements, computer output, and code examples.

*Constant width italic*
Indicates placeholders (for which you substitute an actual name) in examples and in registry keys.

*Constant width bold*
Indicates user input.
Indicates a tip, suggestion, or general note. For example, we’ll tell you when you need to use a particular version or whether an operation requires certain privileges.

Indicates a warning or caution. For example, we’ll tell you when Active Directory does not behave as you’d expect or whether a particular operation has a negative impact on performance.

**Method, Property, Field Name Qualification**

When introduced for the first time, the methods are qualified with their class names; for example, when you see the FindAddress method from the FindServiceSoap, you see it as the FindServiceSoap.FindAddress method, while in subsequent sections you see it as only FindAddress.

**Code Samples**

All code samples are presented in C#. Many code samples and snippets are not wrapped in try/catch blocks for the sake of simplicity; however, it is good practice to wrap the application logic in try/catch blocks to avoid unexpected errors.

**Companion Material**

This book comes with companion material that includes sample code for:

**MapPoint 2004**

All code samples have references to MapPoint 2004; however, since future releases will be backward-compatible with MapPoint 2004, you should not have issues when running samples on future versions of MapPoint.

**MapPoint Web Service**

Samples require credentials. The companion material contains shared credentials for the MapPoint Web Service staging environment. Please note that these credentials are only included for the sake of convenience, and it is recommended to request your own credentials when you start developing with MapPoint Web Service. Abuse of these credentials may result in denied access to the Web Service, causing inconvenience to your fellow readers of this book.

**MSN Virtual Earth**

Sample code uses the undocumented Virtual Earth APIs.

**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 O’Reilly for permission unless you are producing 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: “Programming MapPoint in .NET by Chandu Thota. Copyright © 2006 O’Reilly Media, Inc., 0-596-00906-2.”

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

Other Resources

Writing a technical book is never complete, especially when four different products and technologies are covered in one book. For more information, you can always go to the MSDN online developer center for all your documentation needs. You can read about all of the MapPoint products at http://www.msdn.com/mappoint.

You can also check my MSDN weblog at http://blogs.msdn.com/cthota.

Safari® Enabled

When you see a Safari® Enabled icon on the cover of your favorite technology book, it means the book is available online through the O’Reilly Network Safari Bookshelf.

Safari offers a solution that’s better than e-books. It’s a virtual library that lets you easily search thousands of top technology books, cut and paste code samples, download chapters, and find quick answers when you need the most accurate, current information. Try it for free at http://safari.oreilly.com.

We’d Like Your Feedback!

The information in this book has been tested and verified to the best of our ability, but mistakes and oversights do occur. Please let us know about errors you may find, as well as your suggestions for future editions, by writing to:

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)
You also can send us messages using email. To be put on our mailing list or to request a catalog, send email to:

info@oreilly.com

To ask technical questions or comment on the book, send email to:

bookquestions@oreilly.com

For corrections and amplifications to this book, check out O’Reilly Media’s online catalog at:

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

Acknowledgments

Thanks to O’Reilly for signing this book. I’d also like to thank my editor, Simon St. Laurent, for being patient with me and providing words of encouragement throughout the process.

Thanks to the MapPoint team for creating a fantastic set of products to write about!

Thanks to Wayne S. Freeze, Michael Schmalz, and Dylan Vance for serving as the technical reviewers for this book. All of them caught numerous oversights and mistakes and made the book much better as a result. I’d like especially to thank Dylan’s thoughtful and thorough feedback.

I would like to thank Stephen Lawler for writing the foreword and laying out the vision for the future of MapPoint Development Platform for this book. I also would like to extend my special thanks to Anurag Sharma and Jay Nanduri for their encouragement and cooperation throughout this effort.

Writing a book is a collective effort and it was simply not possible to finish this project without help from the following people: Amit Dekate, Andrew Hwangbo, Brian Jepson, Caleb Thompson, Chris Pendleton, David Buerer, Eric Frost, Norm Bryar, Rachel Falzone, Richard Waymire, Stuart Macrae, and Steve Lombardi.

Finally, I would like to thank to my wife, Taarinya, for putting up with me when I essentially ignored the world writing this book through many weeks, long weekends, and late nights.
So, you want to develop location-based applications using .NET! Microsoft MapPoint technologies offer a wide variety of applications, services, and tools to enable you to develop powerful location-based applications using .NET technologies.

In this introductory chapter, I will go through different location-based application categories and architectures and explain which MapPoint product or technology is appropriate to use in certain scenarios; specifically, I will discuss the fundamental differences between the following three products and technologies from MapPoint:

- MapPoint 2004
- MapPoint Web Service
- MapPoint Location Server

Location-Based Application Categories

Fundamentally, location-based applications are applications that either know how to process location-based information or make use of their location for other processing. To that end, location-based applications can be categorized into two major categories: location-enabled applications and location-aware (or real-time) applications.

Location-Enabled Applications

Location-enabled applications understand location and know how to process it. For example, a conventional store locator is a location-enabled application—simply specify a location and provide a distance within which you want to find stores. Another example is a simple maps-and-directions application that can calculate driving directions using starting and ending addresses and display them on a map. These applications know how to interpret and process the location information.
About the Author

Chandu Thota is Microsoft’s point man on MapPoint. You can see his picture at http://msdn.microsoft.com/mappoint. His weblog at http://blogs.msdn.com/cthota is a key resource for MapPoint developers.

Colophon

Our look is the result of reader comments, our own experimentation, and feedback from distribution channels. Distinctive covers complement our distinctive approach to technical topics, breathing personality and life into potentially dry subjects.

The animal on the cover of Programming MapPoint in .NET is a great frigate bird (fregata minor), also known as a man-of-war bird. Once thought of as a bad omen by Europeans settling in the Western Hemisphere, frigate birds are indeed unpopular with other tropical birds, whose prey they steal with their long, hooked beaks.

However, frigate birds are not bullies simply for the pleasure of harassing other birds. Although their diet consists solely of fish, their disproportionately small feet and lack of waterproofing oil on their feathers make catching fish difficult. Although their relatively large wingspan and light body weight enable them to swoop low to the ocean surface to snatch fish, their main food supply comes from chasing other birds and attacking them until they relinquish their fish, either by dropping fresh kills from their beaks or by regurgitating their recently eaten meals, both of which frigate birds will greedily gobble up.

Despite their inability to swim, frigate birds are tropical sea birds that come to land only to breed. About the size of a large chicken, they weigh only four pounds and have a wingspan of up to six feet, so they can fly for extremely long distances, sometimes staying on ocean updrafts for days. During the breeding season, males attract potential mates by puffing up their bare, bright red throat skin to the size of a human head, sometimes larger than the birds themselves.

While both parents tend the nest, males breed every year—twice as frequently as females, since eggs take nearly two months to hatch, and chicks are dependent on their mothers for food for a year and a half after hatching. Although frigate birds are not ready to breed until they are nine years old, they still have plenty of opportunities, since they have an average life span of 30 years.

Reba Libby was the production editor and the copyeditor for Programming MapPoint in .NET. Jeffrey Liggett proofread the book. Claire Cloutier and Adam Witwer provided quality control. Lucie Haskins wrote the index.

Karen Montgomery designed the cover of this book, based on a series design by Edie Freedman. The cover image is from Cassell's Natural History. Karen Montgomery produced the cover layout with Adobe InDesign CS using Adobe’s ITC Garamond font.
David Futato designed the interior layout. This book was converted by Keith Fahlgren to FrameMaker 5.5.6 with a format conversion tool created by Erik Ray, Jason McIntosh, Neil Walls, and Mike Sierra that uses Perl and XML technologies. The text font is Linotype Birka; the heading font is Adobe Myriad Condensed; and the code font is LucasFont’s TheSans Mono Condensed. The illustrations that appear in the book were produced by Robert Romano, Jessamyn Read, and Lesley Borash using Macromedia FreeHand MX and Adobe Photoshop CS. The tip and warning icons were drawn by Christopher Bing. This colophon was written by Reba Libby.