Wrox Programmer to Programmer™

Includes Scott Guthrie's
NerdDinner.com
ASP.NET MVC Walkthrough

Professional

ASP.NET MVC 1.0

Rob Conery, Scott Hanselman, Phil Haack, Scott Guthrie

Updates, source code, and Wrox technical support at **www.wrox.com**

Programmer to Programmer™

Get more out of WROX.com

Interact

Take an active role online by participating in our P2P forums

Chapters on Demand

Purchase individual book chapters in pdf format

Wrox Online Library

Hundreds of our books are available online through Books24x7.com

Join the Community

Sign up for our free monthly newsletter at newsletter.wrox.com

Wrox Blox

Download short informational pieces and code to keep you up to date and out of trouble!

Browse

Ready for more Wrox? We have books and e-books available on .NET, SQL Server, Java, XML, Visual Basic, C#/ C++, and much more!

Contact Us.

We always like to get feedback from our readers. Have a book idea?
Need community support? Let us know by e-mailing **wrox-partnerwithus@wrox.com**

Professional ASP.NET MVC 1.0

Professional
ASP.NET MVC 1.0

Rob Conery
Scott Guthrie
Phil Haack
Scott Hanselman

Wiley Publishing, Inc.

Professional ASP.NET MVC 1.0

Published by
Wiley Publishing, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com.

Published simultaneously in Canada
ISBN: 978-0-470-38461-9
Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Library of Congress Cataloging-in-Publication Data is available from the publisher.

To my sweet wife Kathy, who inspires me everyday.
— Rob Conery

My wife, Akumi, deserves to have her smiling face on the cover as much as I do, for all her support made this possible. And thanks to Cody for his infectious happiness.
— Phil Haack

Thanks to my wife Mo and my sons Zenzo and Thabo for their unlimited supply of smooches.
— Scott Hanselman

About the Authors

Rob Conery works at Microsoft on the ASP.NET team. He is the creator of SubSonic and was the chief architect of the Commerce Starter Kit (a free, Open Source eCommerce platform for .NET). He lives in Kauai, Hawaii, with his wife and two daughters (Maddy and Ruby).

Scott Guthrie is corporate vice president of Microsoft's .NET Developer Division, where he runs the development teams responsible for delivering Microsoft Visual Studio developer tools and Microsoft .NET Framework technologies for building client and Web applications. A founding member of the .NET project, Guthrie has played a key role in the design and development of Visual Studio and the .NET Framework since 1999. Guthrie is also responsible for Microsoft's web server platform and development tools teams. He has also more recently driven the development of Silverlight — a cross browser, cross platform plug-in for delivering next generation media experiences and rich Internet applications for the Web. Today, Guthrie directly manages the development teams that build the Common Language Runtime (CLR), ASP.NET, Silverlight, Windows Presentation Foundation (WPF), IIS, Commerce Server, and the Visual Studio Tools for web, client, and Silverlight development. Guthrie graduated with a degree in computer science from Duke University.

Phil Haack is a senior program manager with the ASP.NET team working on the ASP.NET MVC project. Prior to joining Microsoft, Phil worked as a product manager for a code search engine, a dev manager for an online gaming company, and a senior architect for a popular Spanish language television network, among other crazy pursuits. As a code junkie, Phil Haack loves to craft software. Not only does he enjoy writing software, but he also enjoys writing about software and software management on his blog, `http://haacked.com`. In his spare time, Phil contributes to various Open Source projects and is the founder of the Subtext blog engine project, which is undergoing a rewrite, using ASP.NET MVC, of course.

Scott Hanselman works for Microsoft as a principal program manager in the Developer Division, aiming to spread the good word about developing software, most often on the Microsoft stack. Before this, he worked in eFinance for 6+ years and before that he was a principal consultant and a Microsoft Partner for nearly 7 years. He was also involved in a few things like the MVP and RD programs and will speak about computers (and other passions) whenever someone will listen to him. He blogs at `www.hanselman.com` and podcasts at `www.hanselminutes.com` and contributes to sites like `www.asp.net`, `www.windowsclient.net`, and `www.silverlight.net`. You can also find him on Twitter, far too often.

Credits

Associate Publisher
Jim Minatel

Development Editor
Maureen Spears

Technical Editors
Levi Broderick
Darren Kindberg

Production Editor
Kathleen Wisor

Copy Editor
Foxxe Editorial Services

Editorial Manager
Mary Beth Wakefield

Production Manager
Tim Tate

Vice President and Executive Group Publisher
Richard Swadley

Vice President and Executive Publisher
Barry Pruett

Project Coordinator, Cover
Lynsey Stanford

Compositor
Craig Woods, Happenstance Type-O-Rama

Proofreader
Nancy C. Hanger, Windhaven

Indexer
J&J Indexing

Acknowledgments

Thanks to my wife for her unflagging support. When Scott Guthrie showed me this "pet project," I told him I just had to work on it, so thanks to The Gu for helping to make that possible. Thanks to Levi Broderick for all his editing help, to Brad Wilson for reviewing the chapter on TDD (I still owe you a beer or two), to Eilon Lipton, the lead developer on ASP.NET MVC, for all his deep insight, and to the rest of the MVC feature team (Carl, Fede, Jon, Keith, Simon etc.) for being so much fun to work with.

— Phil Haack

Thanks to The Gu, and my boss Simon for their support in working on this book. Thanks to Phil Haack, Eilon Lipton, Levi Broderick, and all the ASP.NET MVC guys for making such a rockin' sweet framework.

— Scott Hanselman

Contents

Introduction

Why does the world need Yet Another Web Framework?

This is the question that is most likely on your mind — or perhaps it's what you were thinking when you saw this book sitting on the shelf. We each asked ourselves this many times over the last few years.

Indeed there are many frameworks out there today flavored with every buzzword the industry can think of. In short, it's easy to be skeptical. Yet as we, the authors, delve deeper into the latest and greatest web framework, we're each starting to realize just how far the industry has come in the last 10 years.

Rob began programming for the Web with Classic ASP in 1997 and was giddy with excitement. When .NET came out, he remembers running around his office, stopping everyone from working and explaining that the world just tilted on its axis.

We all feel the same way about ASP.NET MVC. Not because it's "something different" but because it offers developers the ultimate chance to "do it their way." You don't like the way the platform renders the View? Change it! Just about every part of the ASP.NET MVC Framework is "swappable" — if the shoes pinch, get different shoes. Don't like ties? Why not a bow tie? You're totally in control.

ASP.NET MVC is a web framework that comes with a bunch of conventions to make your life easier when you follow them, but if you don't want them, the framework is quick to step out of your way so that you can get your work done in the way you like.

This book is going to go into the "out-of-the-box" experience you'll have with ASP.NET MVC, but more importantly you'll learn practical ways that you can extend ASP.NET MVC with your own magic — then hopefully share that magic with others.

Because of this extensibility and attention to "doing it your way," we're happy to embrace Yet Another Web Framework and hope you are willing to come along with us for the ride.

Who This Book Is For

This book is for web developers who are looking to add more complete testing to their web sites, and who are perhaps ready for "something different."

In some places, we assume that you're somewhat familiar with ASP.NET Web Forms, at least peripherally. There are a lot of ASP.NET Web Forms developers out there who are interested in ASP.NET MVC, so there are a number of places in this book where we contrast the two technologies. Even if you're not already an ASP.NET developer, you might still find these sections interesting for context, as well as for your own edification, as ASP.NET MVC may not be the web technology that you're looking for.

It's worth noting, yet again, that ASP.NET MVC is not a replacement for ASP.NET Web Forms. Many web developers have been giving a lot of attention to other web frameworks out there (Ruby on Rails,

Django), which have embraced the MVC (Model-View-Controller) application pattern, and if you're one of those developers, or even if you're just curious, this book is for you.

MVC allows for (buzzword alert!) a "greater separation of concerns" between components in your application. We'll go into the ramifications of this later on, but if it had to be said it in a quick sentence: *ASP.NET MVC is ASP.NET Unplugged.* ASP.NET MVC is a tinkerer's framework that gives you very fine-grained control over your HTML and JavaScript, as well as complete control over the programmatic flow of your application.

There are no declarative server controls in MVC, which some people may like, others may dislike. In the future, the MVC team may add declarative view controls to the mix, but these will be far different from the components that ASP.NET Web Forms developers are used to, in which a control encapsulates both the logic to render the view and the logic for responding to user input etc. Having all that encapsulated in a single control in the view would violate the "separation of concerns" so central to this framework. The levels of abstraction have been collapsed, with all the doors and windows opened to let the air flow freely.

The final analogy we can throw at you is that ASP.NET MVC is more of a motorcycle, whereas ASP.NET Web Forms might be more like a minivan, complete with airbags and a DVD player in case you have kids and you don't want them to fight while you're driving to the in-laws for Friday dinner. Some people like motorcycles, some people like minivans. They'll both get you where you need to go, but one isn't technically *better* than the other.

How This Book Is Structured

This book is divided into three very broad sections, each comprising several chapters.

The first third of the book is concerned with introducing the MVC pattern and how ASP.NET MVC implements that pattern.

Chapter 1 starts off with a description of the Model-View-Controller pattern, explaining the basic concepts of the pattern and providing a bit of its history. The chapter goes on to describe the state of the MVC pattern on the Web today as it is implemented by various frameworks, such as ASP.NET MVC.

Chapter 2 covers the ways that ASP.NET MVC is different from ASP.NET Web Forms and how to get ASP.NET MVC up and running.

Chapter 3 explores the structure of a standard MVC application and covers what you get out of the box. It covers some of the conventions and the digs a little under the hood to take a look at the entire request lifecycle for an ASP.NET MVC request.

Chapter 4 digs deep into routing to describe the role that URLs play in your application and how routing figures into that. It also differentiates routing from URL rewriting and covers a bit on extending routing and writing unit tests for routes.

Chapter 5 takes a look at controllers and controller actions — what they are and how to write them. It also covers action results, which are returned by controller actions and what they are used for.

Chapters 6–7 cover views and view engines, and then add a little flavor on top by examining the role that AJAX plays in your views.

The second third of the book focuses entirely on advanced techniques and extending the framework.

Chapter 8 goes into detail on action filters, which provide an extensibility point for adding cross-cutting behaviors to action methods.

Chapter 9 covers security and good practices for building a secure application.

Chapter 10 covers various approaches to building and interacting with different types of services made available over the Web.

Chapter 11 provides a brief introduction to Test Driven Development (TDD) as it applies to ASP.NET MVC. It then goes on to examine real-world patterns and practices for building applications that are testable.

The final part of the book covers guidance and best practices as well as providing a look ahead at the future of the ASP.NET MVC platform.

Chapter 12 goes into detail on how Web Forms and MVC fit together and covers ways to have the two coexist in the same application, as well as how to migrate an app from Web Forms to MVC.

We tried to organize the book in such a way that when you read it in order, each chapter builds on the previous one. If you already familiar with ASP.NET MVC you might skip directly to Chapter 4 and go from there.

What You Need to Use This Book

To use ASP.NET MVC, you'll probably want a copy of Visual Studio. You can use Visual Studio 2008 Web Developer Express SP1 or any of the paid versions of Visual Studio 2008 (such as Visual Studio 2008 Professional). If you're going to use the Web Developer Express edition of Visual Studio, you need to confirm that you're using SP1. ASP.NET MVC requires that you use Web Application Projects (WAPs) rather than Web Site Projects, and this functionality was added in SP1 of Web Developer Express.

You will also need to make sure that you have the .NET Framework 3.5 installed at minimum. The runtime does not require .NET 3.5 SP1 to run.

The following list shows you where to go to download the required software.

- Visual Studio or Visual Studio Express: `www.microsoft.com/vstudio or www.microsoft.com/express`
- ASP.NET MVC: `www.asp.net/mvc`

Conventions

To help you get the most from the text and keep track of what's happening, we've used a number of conventions throughout the book.

Occasionally the product team will take a moment to provide an interesting aside, for bits of trivia, and those will appear in boxes like this:

> ***Product Team Aside:*** **Boxes like this one hold tips, tricks, trivia from the ASP.NET Product Team or some other information that is directly relevant to the surrounding text.**

Tips, hints and tricks to the current discussion are offset and placed in italics like this.

As for styles in the text:

- We *highlight* new terms and important words when we introduce them.
- We show keyboard strokes like this: Ctrl+A.
- We show file names, URLs, and code within the text like so: `persistence.properties`.
- We present code in two different ways:

```
In code examples, we highlight important code that we want to emphasize with a gray
background.
```

```
The gray highlighting is not used for code that's less important in the present
context, or has been shown before.
```

Source Code

The main nerddinner.com code download is hosted at codeplex and the most up-to-date code will always be available at `http://www.codeplex.com/nerddinner`. The original nerddinner.com code that matches the code used in the book is hosted at wrox.com from the book page.

As you work through the examples in this book, you may choose either to type in all the code manually or to use the source code files that accompany the book. All of the source code used in this book is available for downloading at `www.wrox.com`. Once at the site, simply locate the book's title (either by using the Search box or by using one of the title lists) and click the Download Code link on the book's detail page to obtain all the source code for the book.

Because many books have similar titles, you may find it easiest to search by ISBN; this book's ISBN is 978-0-470-38461-9.

Once you download the code, just decompress it with your favorite compression tool. Alternately, you can go to the main Wrox code download page at `www.wrox.com/dynamic/books/download.aspx` to see the code available for this book and all other Wrox books.

Errata

We make every effort to ensure that there are no errors in the text or in the code. However, no one is perfect, and mistakes do occur. If you find an error in one of our books, like a spelling mistake or faulty piece of code, we would be very grateful for your feedback. By sending in errata you may save another reader hours of frustration, and at the same time you will be helping us provide even higher-quality information.

To find the errata page for this book, go to `www.wrox.com` and locate the title using the Search box or one of the title lists. Then, on the book details page, click the Book Errata link. On this page, you can view all errata that has been submitted for this book and posted by Wrox editors. A complete book list including links to each book's errata is also available at `www.wrox.com/misc-pages/booklist.shtml`.

If you don't spot "your" error on the Book Errata page, go to `www.wrox.com/contact/techsupport.shtml` and complete the form there to send us the error you have found. We'll check the information and, if appropriate, post a message to the book's errata page, and fix the problem in subsequent editions of the book.

p2p.wrox.com

For author and peer discussion, join the P2P forums at `p2p.wrox.com`. The forums are a Web-based system for you to post messages relating to Wrox books and related technologies and interact with other readers and technology users. The forums offer a subscription feature to e-mail you topics of interest of your choosing when new posts are made to the forums. Wrox authors, editors, other industry experts, and your fellow readers are present on these forums.

At `http://p2p.wrox.com` you will find a number of different forums that will help you not only as you read this book but also as you develop your own applications. To join the forums, just follow these steps:

1. Go to `p2p.wrox.com`, and click the Register link.
2. Read the terms of use, and click Agree.
3. Complete the required information to join as well as any optional information you wish to provide, and click Submit.
4. You will receive an e-mail with information describing how to verify your account and complete the joining process.

You can read messages in the forums without joining P2P, but in order to post your own messages, you must join.

Once you join, you can post new messages and respond to messages other users post. You can read messages at any time on the Web. If you would like to have new messages from a particular forum e-mailed to you, click the Subscribe to this Forum icon by the forum name in the forum listing.

For more information about how to use the Wrox P2P, be sure to read the P2P FAQs for answers to questions about how the forum software works as well as many common questions specific to P2P and Wrox books. To read the FAQs, click the FAQ link on any P2P page.

1

NerdDinner

The best way to learn a new framework is to build something with it. This first chapter walks through how to build a small, but complete, application using ASP.NET MVC, and introduces some of the core concepts behind it.

The application we are going to build is called "NerdDinner." NerdDinner provides an easy way for people to find and organize dinners online (Figure 1-1).

NerdDinner enables registered users to create, edit and delete dinners. It enforces a consistent set of validation and business rules across the application (Figure 1-2).

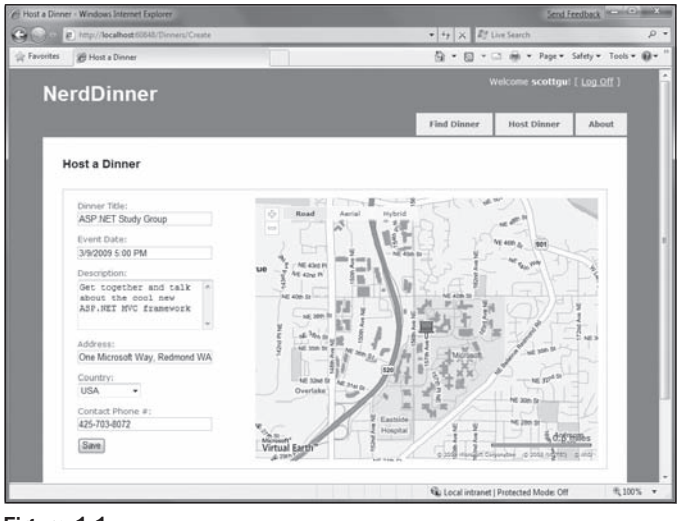

Figure 1-1

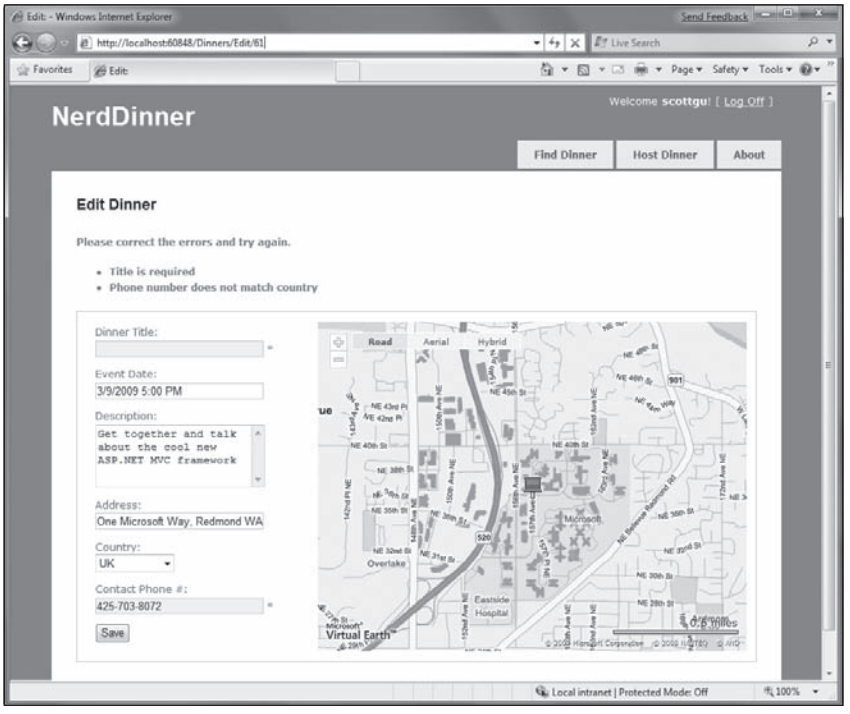

Figure 1-2

Visitors to the site can search to find upcoming dinners being held near them (Figure 1-3):

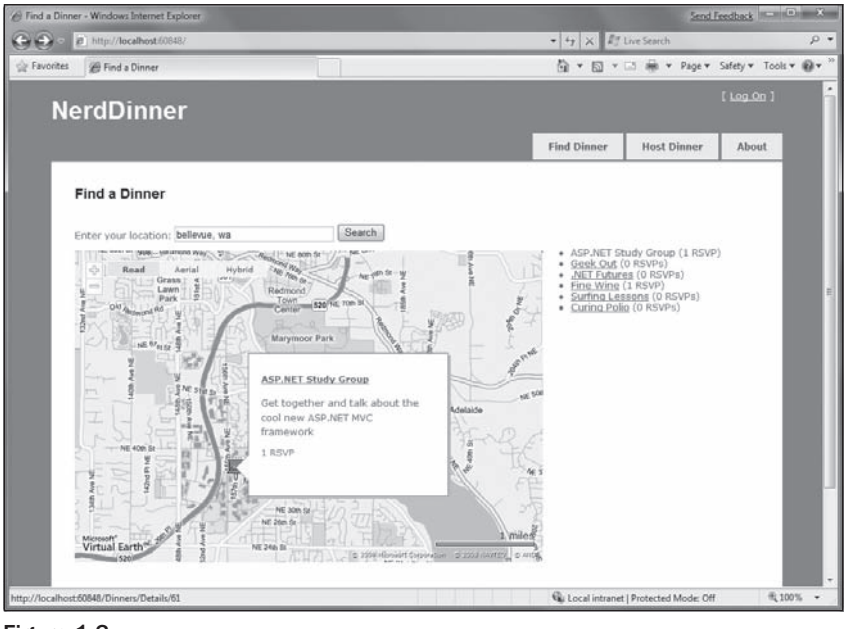

Figure 1-3

Clicking a dinner will take them to a details page where they can learn more about it (Figure 1-4):

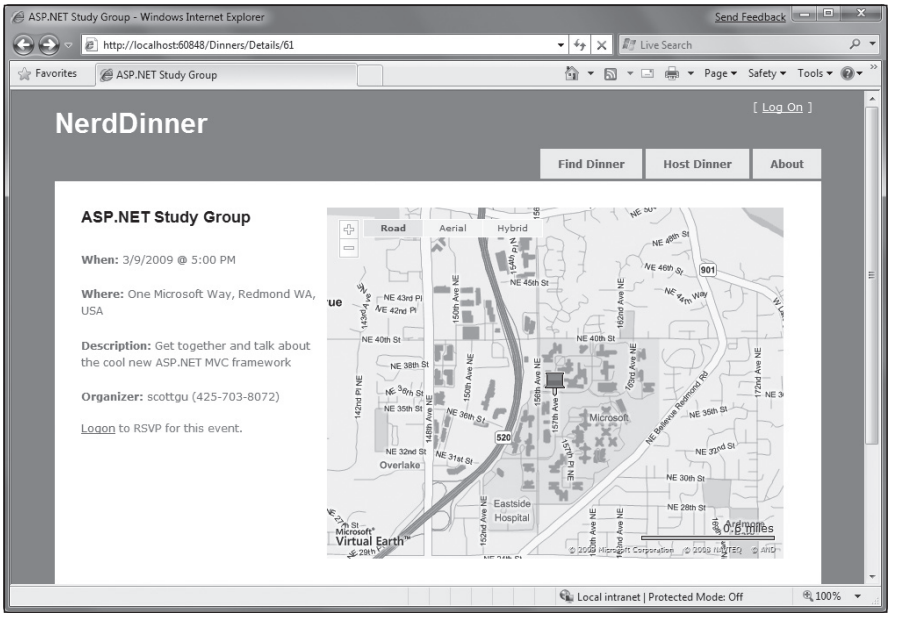

Figure 1-4

If they are interested in attending the dinner they can log in or register on the site (Figure 1-5):

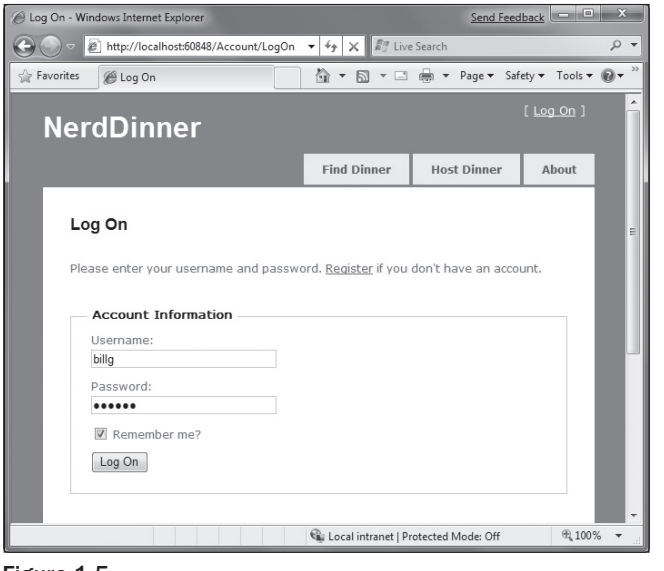

Figure 1-5

They can then easily RSVP to attend the event (Figures 1-6 and 1-7):

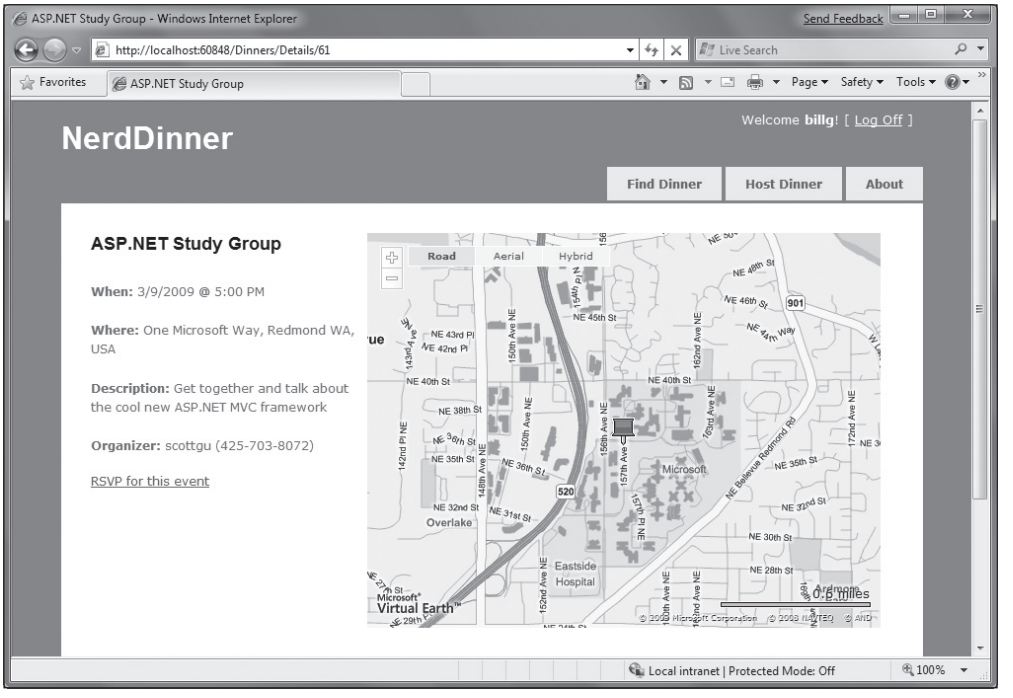

Figure 1-6

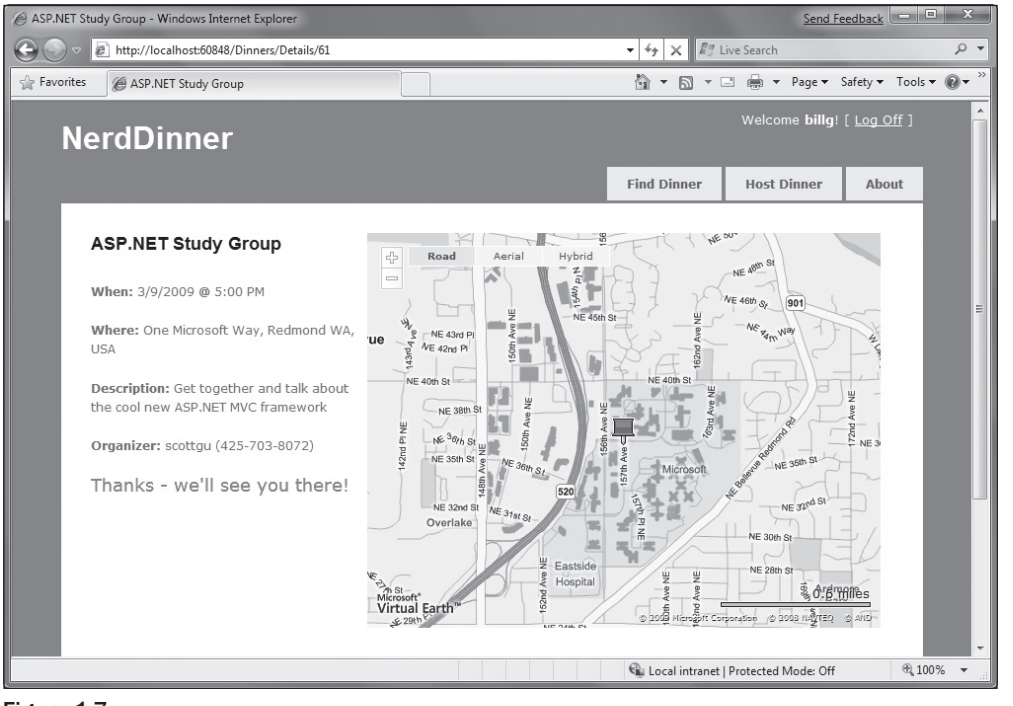

Figure 1-7

We are going to begin implementing the NerdDinner application by using the File ➪ New Project command within Visual Studio to create a brand new ASP.NET MVC project. We'll then incrementally add functionality and features. Along the way we'll cover how to create a database, build a model with business rule validations, implement data listing/details UI, provide CRUD (Create, Update, Delete) form entry support, implement efficient data paging, reuse the UI using master pages and partials, secure the application using authentication and authorization, use AJAX to deliver dynamic updates and interactive map support, and implement automated unit testing.

You can build your own copy of NerdDinner from scratch by completing each step we walk through in this chapter. Alternatively, you can download a completed version of the source code here: `http://tinyurl.com/aspnetmvc`.

You can use either Visual Studio 2008 or the free Visual Web Developer 2008 Express to build the application. You can use either SQL Server or the free SQL Server Express to host the database.

You can install ASP.NET MVC, Visual Web Developer 2008, and SQL Server Express using the Microsoft Web Platform Installer available at `www.microsoft.com/web/downloads`.

File ➪ New Project

We'll begin our NerdDinner application by selecting the File ➪ New Project menu item within Visual Studio 2008 or the free Visual Web Developer 2008 Express.

This will bring up the New Project dialog. To create a new ASP.NET MVC application, we'll select the Web node on the left side of the dialog and then choose the ASP.NET MVC Web Application project template on the right (Figure 1-8):

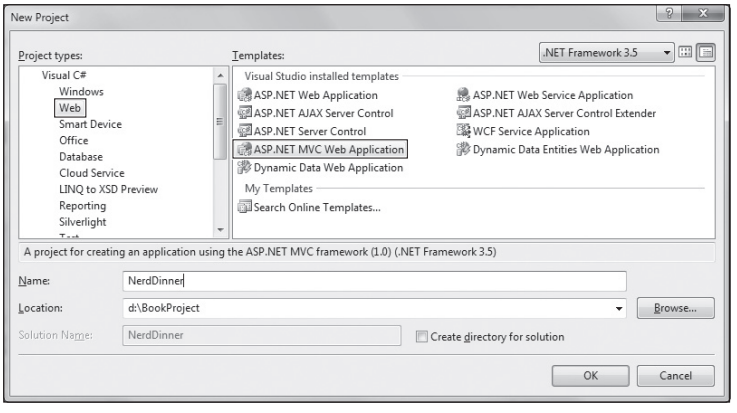

Figure 1-8

We'll name the new project **NerdDinner** and then click the OK button to create it.

When we click OK, Visual Studio will bring up an additional dialog that prompts us to optionally create a unit test project for the new application as well (Figure 1-9). This unit test project enables us to create automated tests that verify the functionality and behavior of our application (something we'll cover later in this tutorial).

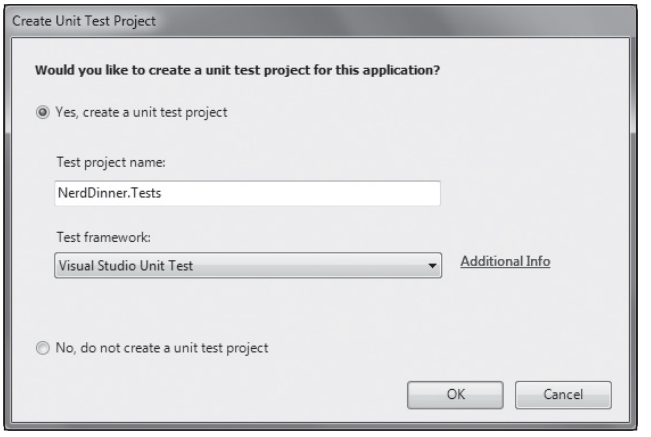

Figure 1-9

The Test framework drop-down in Figure 1-9 is populated with all available ASP.NET MVC unit test project templates installed on the machine. Versions can be downloaded for NUnit, MBUnit, and XUnit. The built-in Visual Studio Unit Test Framework is also supported.

> *The Visual Studio Unit Test Framework is only available with Visual Studio 2008 Professional and higher versions). If you are using VS 2008 Standard Edition or Visual Web Developer 2008 Express, you will need to download and install the NUnit, MBUnit, or XUnit extensions for ASP.NET MVC in order for this dialog to be shown. The dialog will not display if there aren't any test frameworks installed.*

We'll use the default `NerdDinner.Tests` name for the test project we create, and use the Visual Studio Unit Test Framework option. When we click the OK button, Visual Studio will create a solution for us with two projects in it — one for our web application and one for our unit tests (Figure 1-10):

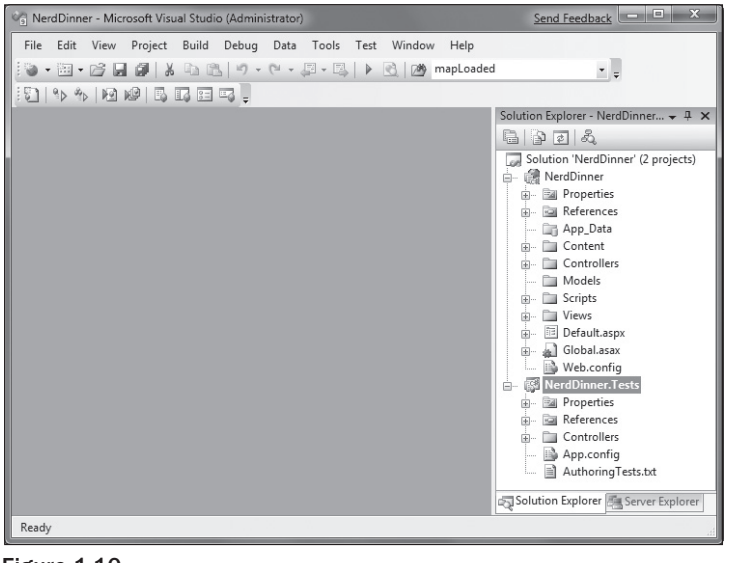

Figure 1-10