CHAPTER 13

# Deployment

The final, and critical, step in application development is deployment: making your application available for the world to use. In this chapter, I'll show you how to prepare and deploy the SportsStore application.

There are lots of different ways to deploy MVC Framework applications and a wide range of deployment targets. You can deploy to a Windows Server machine running Internet Information Services (IIS) which you run and manage locally; you can deploy to a remote hosting service that manages servers for you; or, increasingly, you can deploy to a cloud infrastructure platform that provisions and scales your application to seamlessly meet demand.

I thought long and hard about how to create a useful example deployment in this chapter. I ruled out showing you how to deploy directly to IIS because the server configuration process is long and complicated, and most MVC Framework developers that are targeting local servers rely on an IT operations group to perform configuration and deployment tasks. I also ruled out demonstrating deployment to a managed hosting company because each has its own custom deployment processes and no one company sets the standard for hosting.

So, somewhat by default, I settled on demonstrating a deployment to Windows Azure, which is Microsoft's cloud platform and which has some nice support for MVC applications. I am not suggesting that Azure is suitable for all deployments, but I like the way it works and using it in this chapter allows me to demonstrate the deployment process rather than getting bogged down in IIS and Windows configuring issues. There is a free 90-day trial available on Azure as I write this (and MSDN subscriptions include Azure), which means that you should be able to follow the example in this chapter, even if you don't intend to use Azure to host your application.

---

**Caution** I recommend that you practice deployment using a test application and server before attempting to deploy a real application into a production environment. Like every other aspect of the software development life cycle, the deployment process benefits from testing. I have a stock of horror stories of project teams who have destroyed operational applications through overly hasty and poorly tested deployment procedures. It is not that the ASP.NET deployment features are especially dangerous—they are not—but rather, any interaction that involves a running application with real user data deserves careful thought and planning.

---

Deploying a web application used to be a tedious and error-prone process, but Microsoft has put a lot of effort into improving the deployment tools in Visual Studio. So even if you need to deploy to a different kind of infrastructure, you will find that Visual Studio is able to do a lot of the heavy lifting for you.

## Preparing Windows Azure

You have to create an account before you can use Azure, which you can do by going to `www.windowsazure.com`. At the time of writing, Microsoft is offering free trial accounts, and MSDN packages include Azure services. Once you have created your account, you can manage your Azure services by going to `http://manage.windowsazure.com` to provide your credentials. When you start, you will see the summary view shown in Figure 13-1.

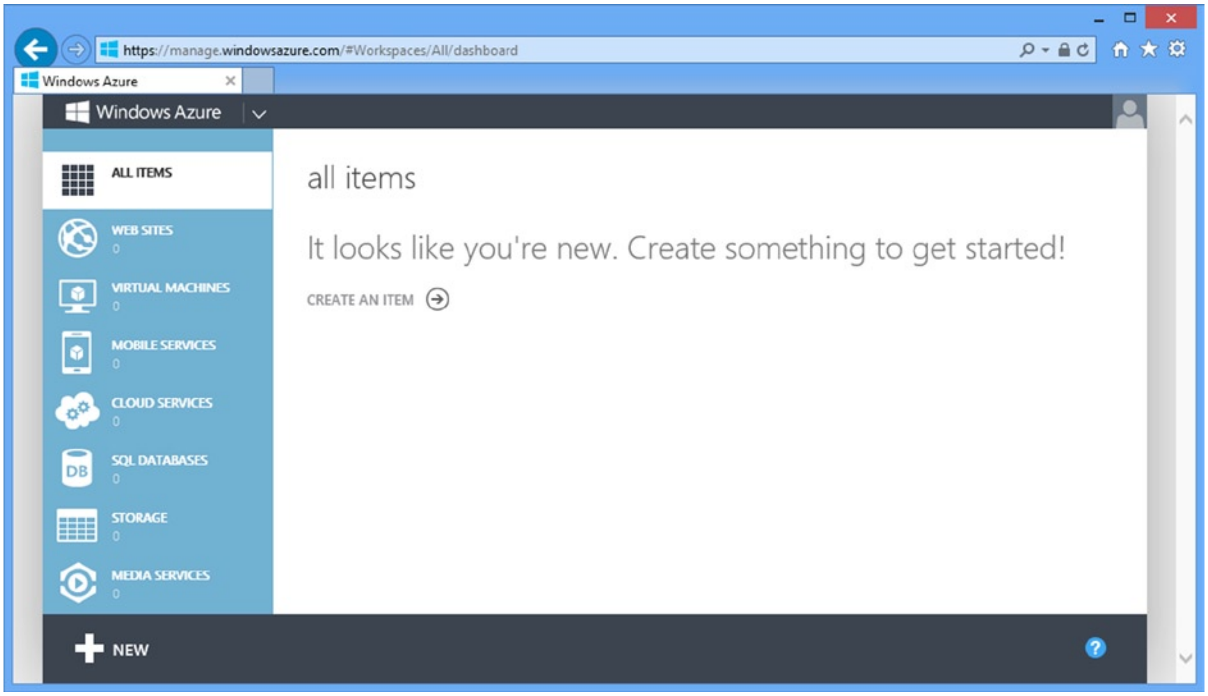
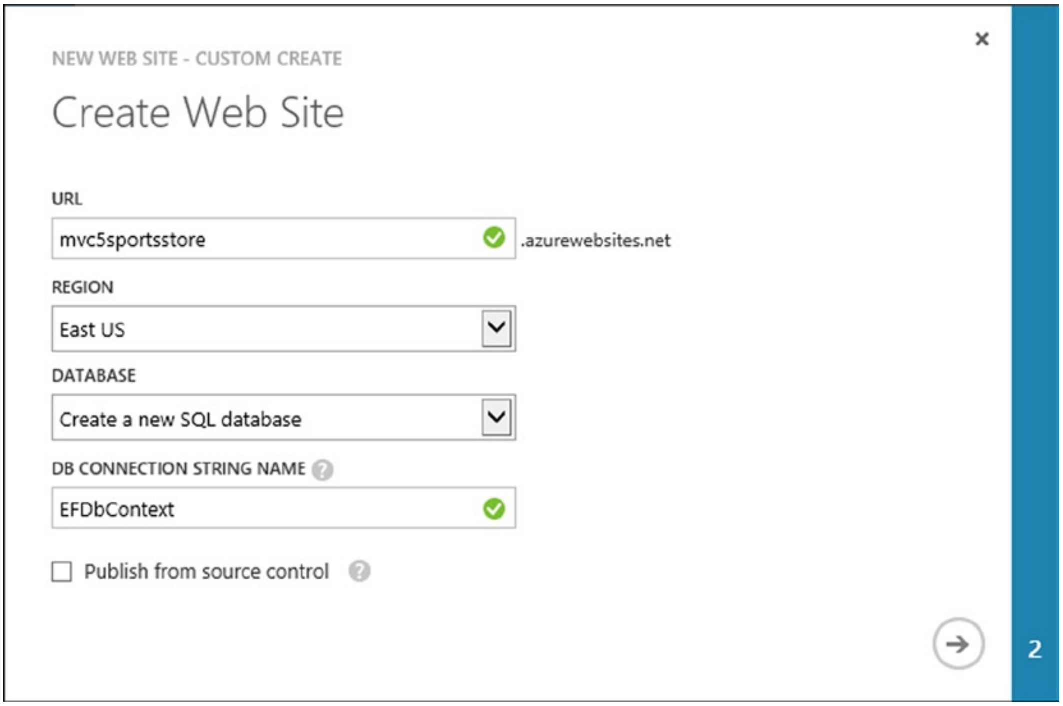


***Figure 13-1.*** *The Azure portal*

### Creating the Web Site and Database

I start by creating a new web site and database service, which are two of the cloud services offered by Azure. Click the large plus sign in the bottom-left corner of the portal window and select `Compute` ➤ `Web Site` ➤ `Custom Create`. You will see the form illustrated in Figure 13-2.


NEW WEB SITE - CUSTOM CREATE

Create Web Site

URL: mvc5sportsstore .azurewebsites.net

REGION: East US

DATABASE: Create a new SQL database

DB CONNECTION STRING NAME: EFDbContext

☐ Publish from source control

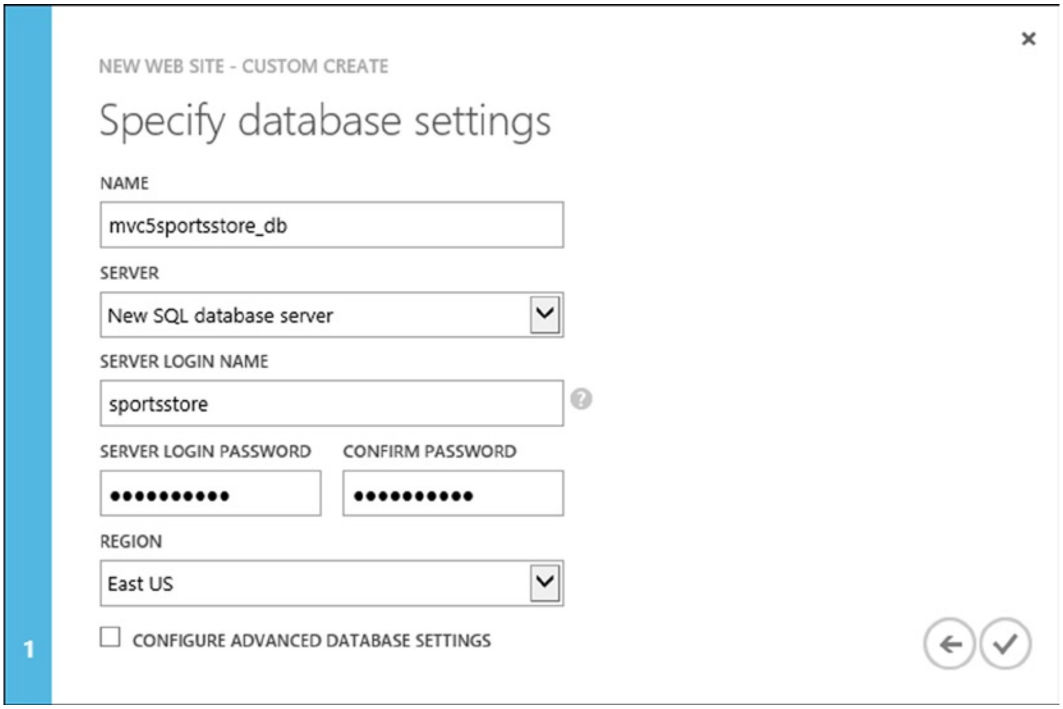
*Figure 13-2.* *Creating a new web site and database*

I need to select a URL for my application. For the free Azure services, I am restricted to names in the `azurewebsites.net` domain. I have chosen the name `mvc5sportsstore`, but you will have to choose your own name since each Azure web site requires a unique name.

Select the region that you want your application deployed to and ensure that the `Create a new SQL database` option is selected for the `Database` field. (Azure can use MySql, which the SportsStore application is not set up to use, so I want the option that gives me a SQL Server database.)

Set the DB `Connection String Name` field to `EFDbContext`. This is the name the `SportsStore` application uses to get a database connection from the `Web.config` file, and by using this name in the Azure service, I ensure that the application code works in deployment without modification.

When you have filled out the form, click the arrow button to proceed to the form shown in Figure 13-3.


NEW WEB SITE - CUSTOM CREATE

Specify database settings

NAME: mvc5sportsstore_db

SERVER: New SQL database server

SERVER LOGIN NAME: sportsstore

SERVER LOGIN PASSWORD: ●●●●●●●●●●●

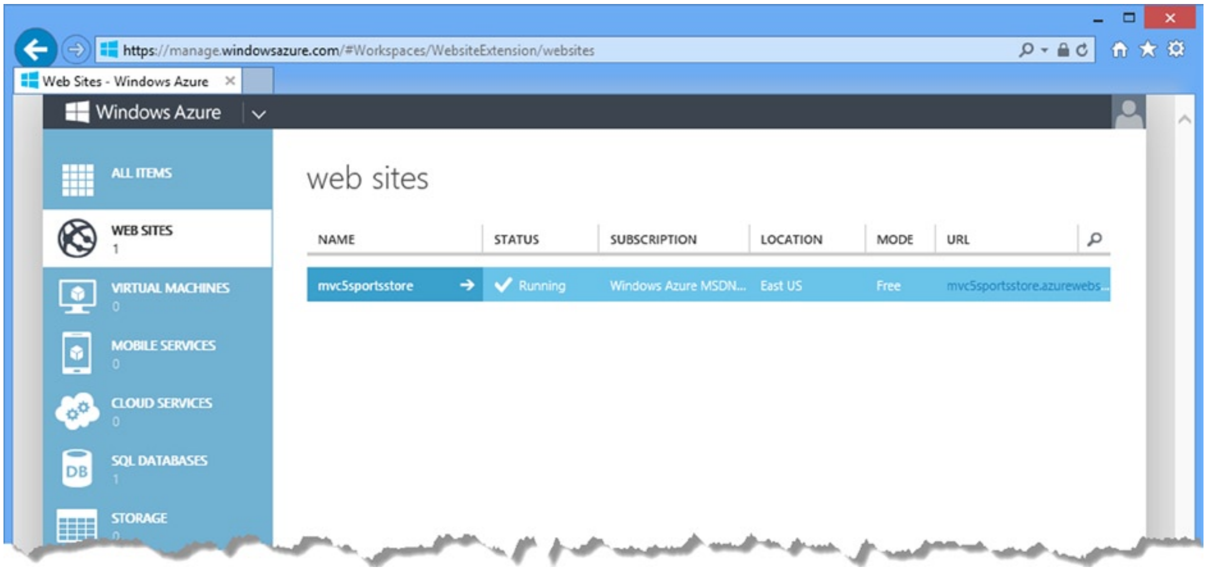
CONFIRM PASSWORD: ●●●●●●●●●●●

REGION: East US

☐ CONFIGURE ADVANCED DATABASE SETTINGS

*Figure 13-3.* *Configuring the database*

Set a name for the database. I have used `mvc5sportsstore_db` so that it is obvious which application the database relates to. Select the `New SQL Data Server` option for the `Server` field and enter a login name and password. I specified a name of `sportsstore` and followed the guidance provided by the form to select a password containing mixed-case letters and numbers. Make a note of the username and password you use because you will need them in the next section. Click the check mark button to complete the setup process. Azure will create new web site and database services, which can take a few minutes. You will be returned to the overview when setup is complete and you will see that the `Web Sites` and `SQL Databases` categories each report one item, as shown in Figure 13-4.



| NAME | STATUS | SUBSCRIPTION | LOCATION | MODE | URL |
|---|---|---|---|---|---|
| mvc5sportsstore | ✓ Running | Windows Azure MSDN... | East US | Free | mvc5sportsstore.azurewebs... |

*Figure 13-4.* *The effect of creating a web site with a database*

## Preparing the Database for Remote Administration

The next step is to configure the Azure database so that it contains the same schema and data that I used in Chapter 7. Click the `SQL Databases` link in the Azure summary page and then click the entry that appears in the `SQL Databases` table. (If you are following my example, the database will be called `mvc5sportsstore_db`.)

The portal will show you details of the database and various options for configuring and managing it. Click the `Set up Windows Azure firewall rules for this address` link, which you will find in the `Design Your Database` section of the page. You will see a message that tells you that your current IP address (which is to say the IP address of your workstation) is not in the firewall rules. Click the `Yes` button as shown in Figure 13-5.

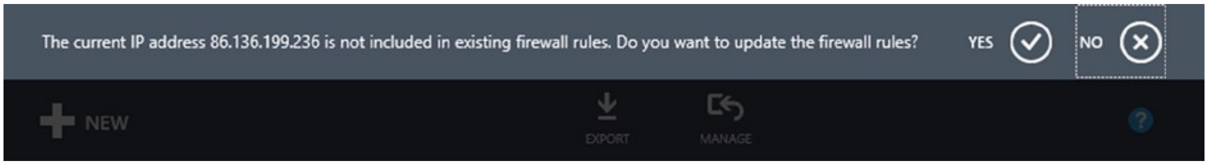
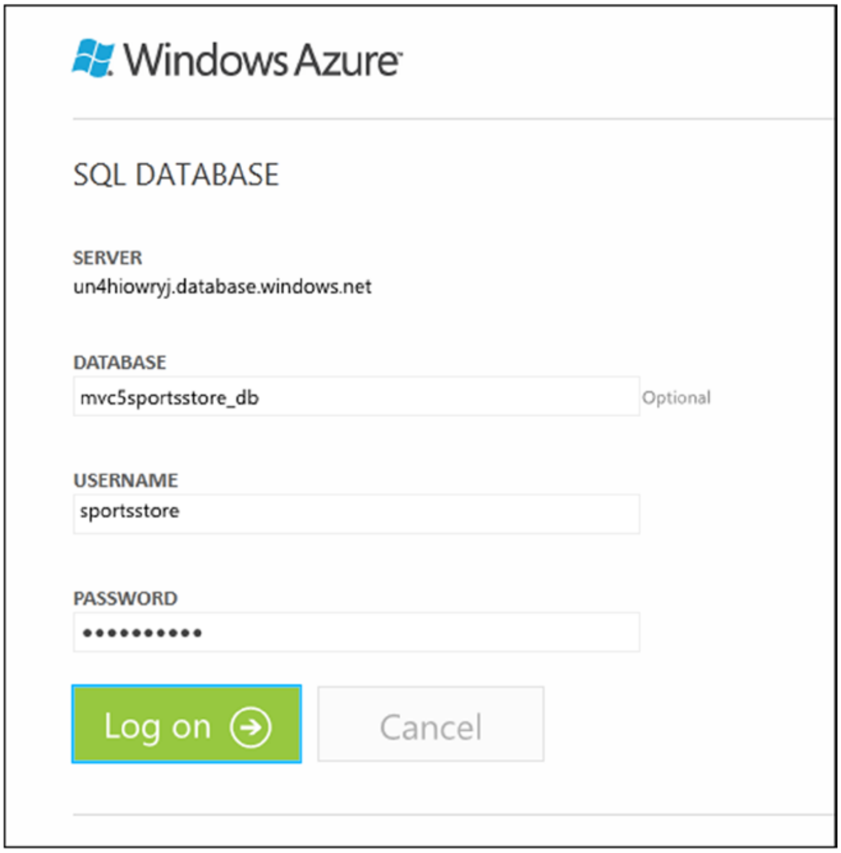

The current IP address 86.136.199.236 is not included in existing firewall rules. Do you want to update the firewall rules?

*Figure 13-5.* *Adding the workstation IP address to the Azure firewall rules*

■ **Caution** Visual Studio has support for deploying the database along with the application. I recommend against using this feature since it is trivially easy to wipe out your real application data with a careless menu selection. Always update your database separately and test thoroughly before doing so.

## Creating the Schema

The next step is to create the schema for the database. Click the `Design your SQL database` link in the `Connect to your Database` section. Enter the database name (`mvc5sportsstore_db`), username (`sportsstore`), and the password that you defined when creating the database, and click the Log On button, as shown in Figure 13-6.


Windows Azure

SQL DATABASE

SERVER
un4hiowryj.database.windows.net

DATABASE
mvc5sportsstore_db Optional

USERNAME
sportsstore

PASSWORD
●●●●●●●●●●

***Figure 13-6.*** *Connecting to the database*

---

■ **Tip** Managing the database requires Silverlight, which you may need to install in your browser before you can continue.

---

At the top of the window you will see a `New Query` button. Click the button and a text area will enter into which you can type SQL commands. This is where I am going to provide the SQL commands that will create the database table I need.

## Getting the Schema Command

I can get the SQL statements I need from Visual Studio. Open the `Server Explorer` window and expand the items it contains until you reach the entry for the `Products` table in the development SportsStore application. Right-click the `Products` table and select `Open Table Definition`. The editor for the table schema will be opened. In the T-SQL tab, you will see the SQL shown in Listing 13-1.

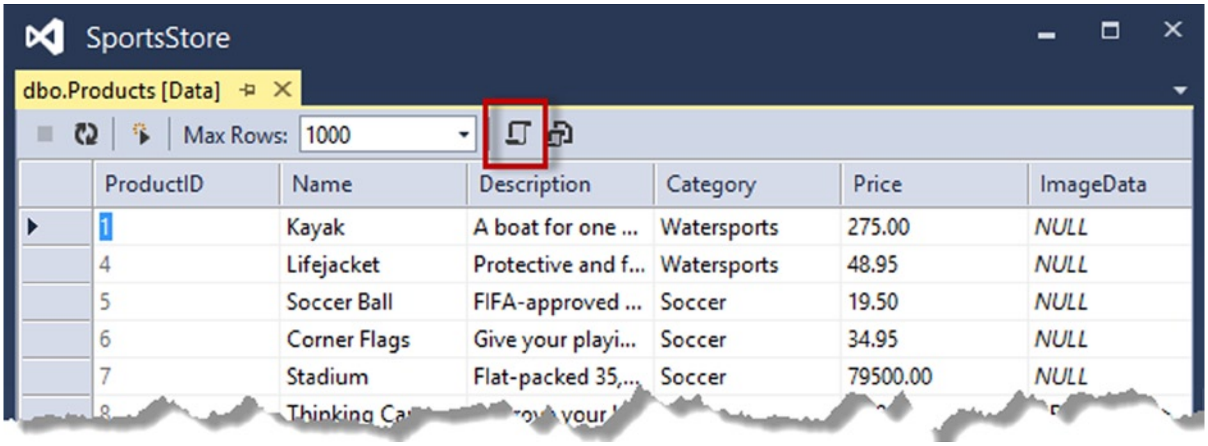
***Listing 13-1.*** The Statement to Create the Products Table

```
CREATE TABLE [dbo].[Products] (
    [ProductID]     INT             IDENTITY (1, 1) NOT NULL,
    [Name]          NVARCHAR (100)  NOT NULL,
    [Description]   NVARCHAR (500)  NOT NULL,
    [Category]      NVARCHAR (50)   NOT NULL,
    [Price]         DECIMAL (16, 2) NOT NULL,
    [ImageData]     VARBINARY (MAX) NULL,
    [ImageMimeType] VARCHAR (50)    NULL,
    PRIMARY KEY CLUSTERED ([ProductID] ASC)
);
```

Copy the SQL from Visual Studio, paste it into the text area in the browser, and click the `Run` button at the top of the browser window. After a second, you will see the message `Command(s) completed successfully`, which indicates that the Azure database contains a `Product` database using the same schema as I defined in the `SportsStore` application.

## Adding the Table Data

Now that I have created the table, I can populate it with the product data that I used in Chapter 7. Return to the `Products` entry in the `Database Explorer` window, right-click, and select `Show Table Data` from the pop-up menu. You will find a `Script` button at the top of the window that is opened, as shown in Figure 13-7.



| ProductID | Name | Description | Category | Price | ImageData |
|---|---|---|---|---|---|
| 1 | Kayak | A boat for one ... | Watersports | 275.00 | NULL |
| 4 | Lifejacket | Protective and f... | Watersports | 48.95 | NULL |
| 5 | Soccer Ball | FIFA-approved ... | Soccer | 19.50 | NULL |
| 6 | Corner Flags | Give your playi... | Soccer | 34.95 | NULL |
| 7 | Stadium | Flat-packed 35,... | Soccer | 79500.00 | NULL |
| 8 | Thinking Ca | your | | | |

***Figure 13-7.*** *The script button in the table data display*

A new window will open containing another SQL statement, which I have shown in Listing 13-2.

***Listing 13-2.*** The SQL Statement to Add Data to the Products Table

```
SET IDENTITY_INSERT [dbo].[Products] ON
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (1, N'Kayak', N'A boat for one person', N'Watersports', CAST(275.00 AS
Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (4, N'Lifejacket', N'Protective and fashionable', N'Watersports', CAST(48.95
AS Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (5, N'Soccer Ball', N'FIFA-approved size and weight', N'Soccer', CAST(19.50
AS Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (6, N'Corner Flags', N'Give your playing field a professional touch',
N'Soccer', CAST(34.95 AS Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (7, N'Stadium', N'Flat-packed 35,000-seat stadium', N'Soccer', CAST(79500.00
AS Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (8, N'Thinking Cap', N'Improve your brain efficiency by 75%', N'Chess',
CAST(16.00 AS Decimal(16, 2)), N'image/jpeg')
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (9, N'Unsteady Chair', N'Secretly give your opponent a disadvantage',
N'Chess', CAST(29.95 AS Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (10, N'Human Chess Board', N'A fun game for the family', N'Chess',
CAST(75.00 AS Decimal(16, 2)), NULL)
INSERT INTO [dbo].[Products] ([ProductID], [Name], [Description], [Category], [Price],
[ImageMimeType]) VALUES (11, N'Bling-Bling King', N'Gold-plated, diamond-studded King', N'Chess',
CAST(1200.00 AS Decimal(16, 2)), NULL)
SET IDENTITY_INSERT [dbo].[Products] OFF
```

Clear the text area in the Azure browser window and paste the SQL shown in the listing in its place. Click the `Run` button. The script will be executed and add the data to the table.

## Deploying the Application

Now that the setup is complete, deploying the application is simple. Return to the main Azure portal and click the `Web Sites` button. Click the `mvc5sportsstore` web site to open the dashboard page and click the `Download the publish profile` link in the `Publish your app` section. Save this file in a prominent location.

For my Azure service, the file is called `mvc5sportsstore.azurewebsites.net.PublishSettings` and I saved it to the desktop. This file contains the details that Visual Studio needs to publish your app to the Azure infrastructure.

Return to Visual Studio and right-click the `SportsStore.WebUI` project in the Solution Explorer and select `Publish` from the pop-up menu. You will see the Publish Web dialog window, as illustrated by Figure 13-8.

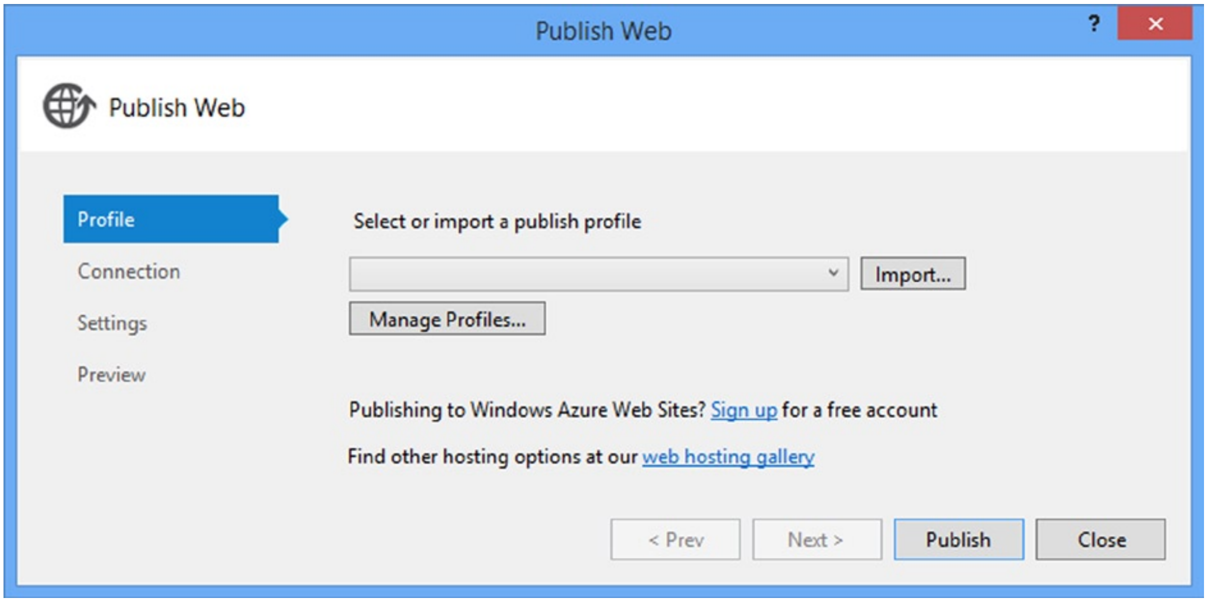
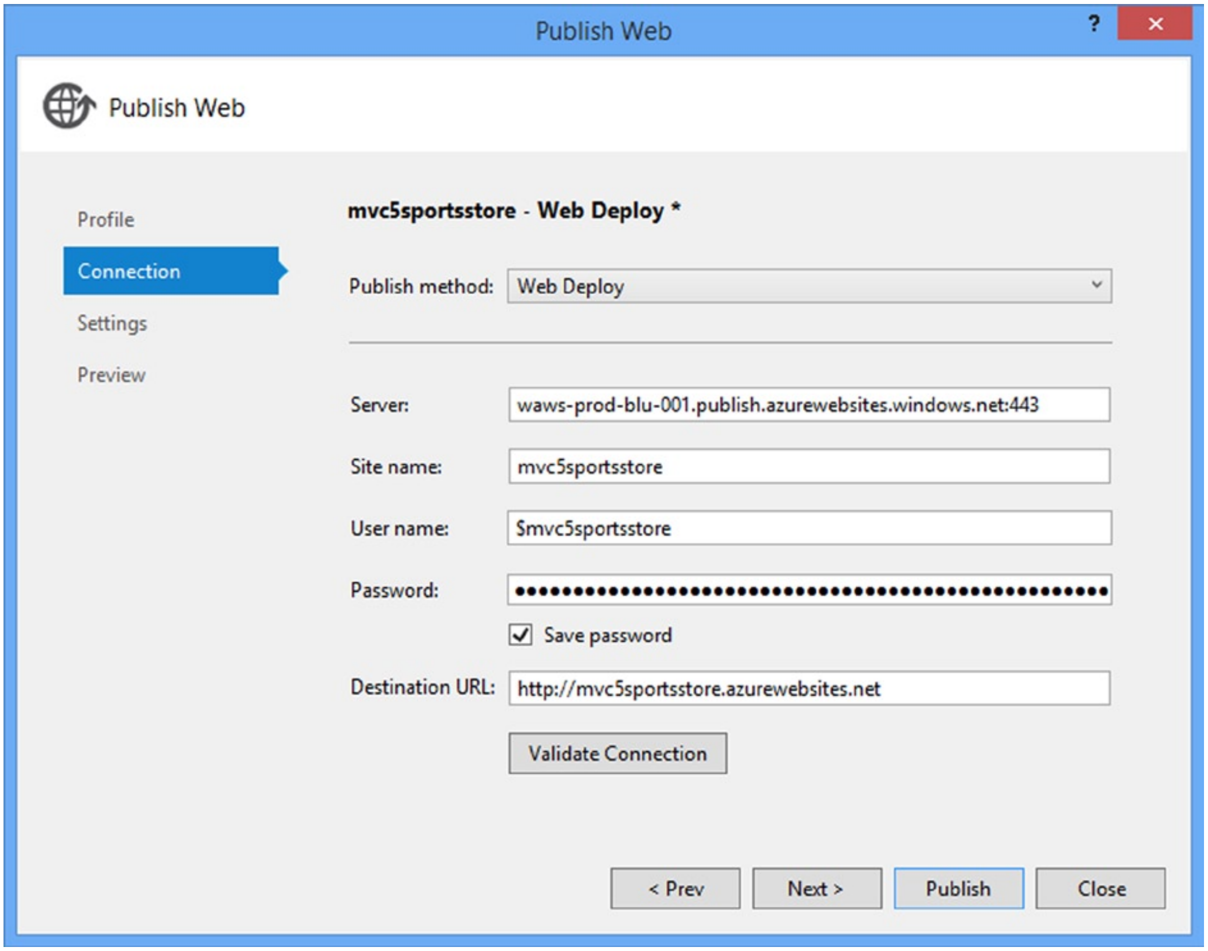

Publish Web
Publish Web
Profile
Connection
Settings
Preview
Select or import a publish profile
Import...
Manage Profiles...
Publishing to Windows Azure Web Sites? Sign up for a free account
Find other hosting options at our web hosting gallery
< Prev
Next >
Publish
Close

***Figure 13-8.*** *The Publish Web dialog*

Click the `Import` button and locate the file that you downloaded from the Azure portal. Visual Studio will process the file and display the details of your Azure service configuration, as shown in Figure 13-9. Your details will reflect the name you selected for your web site.



mvc5sportsstore - Web Deploy *

Publish method: Web Deploy

Server: waws-prod-blu-001.publish.azurewebsites.windows.net:443

Site name: mvc5sportsstore

User name: $mvc5sportsstore

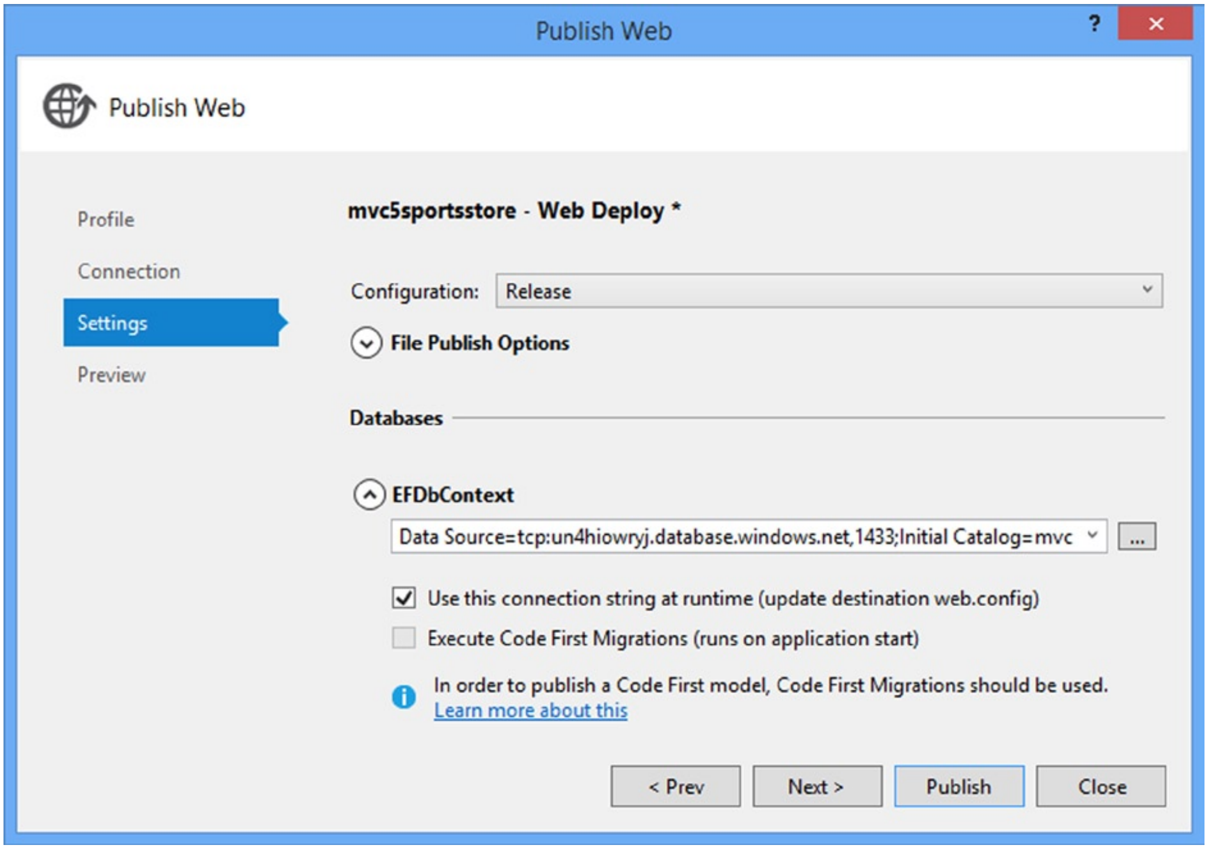
Password: ●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

✓ Save password

Destination URL: http://mvc5sportsstore.azurewebsites.net

***Figure 13-9.*** *Details of the Azure service that the application will be deployed to*

There is no need to change any of the values that are displayed. Click the `Next` button to move to the next stage of the deployment process, which you can see in Figure 13-10.



mvc5sportsstore - Web Deploy *

Configuration: Release

File Publish Options

Databases

EFDbContext

Data Source=tcp:un4hiowryj.database.windows.net,1433;Initial Catalog=mvc

☑ Use this connection string at runtime (update destination web.config)

☐ Execute Code First Migrations (runs on application start)

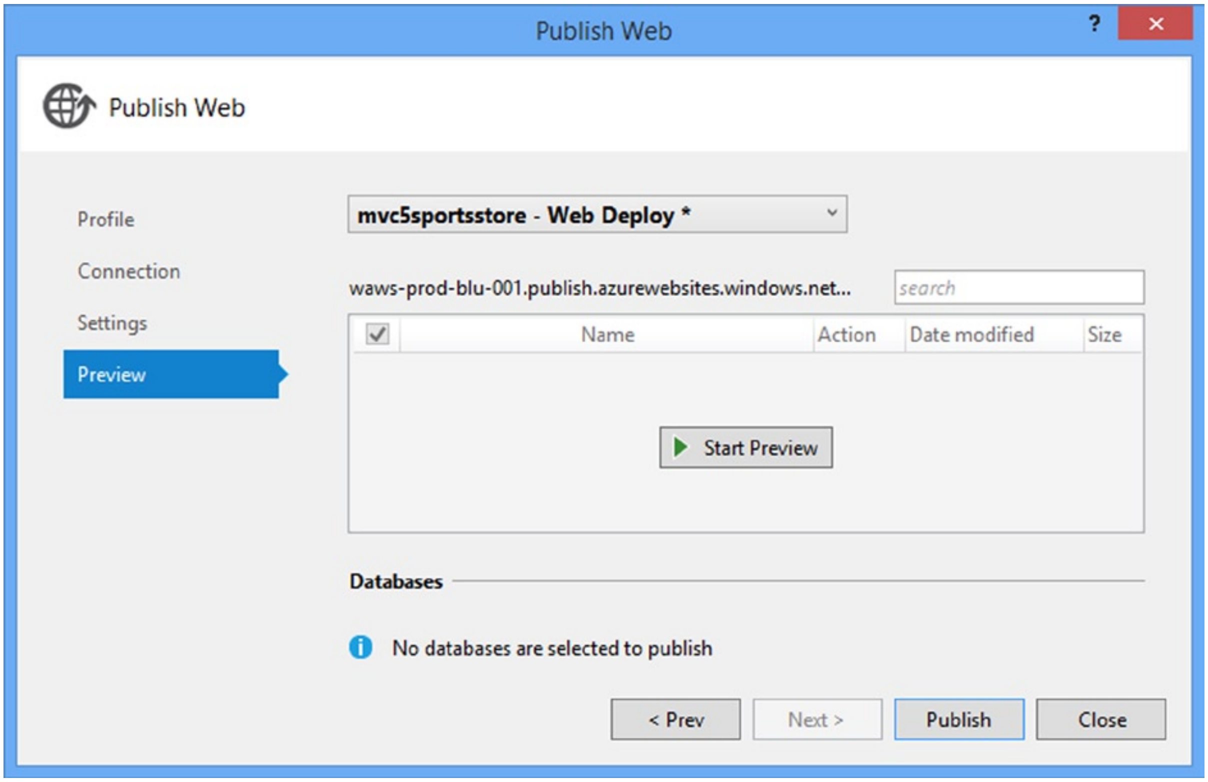
In order to publish a Code First model, Code First Migrations should be used.
Learn more about this

***Figure 13-10.*** *Settings for the deployed application*

You can choose the configuration that will be used in deployment. This will usually be `Release`, but you can select `Debug` if you intend to test your application on the Azure infrastructure and want the debug settings for the compiler and your application bundles.

The other part of this process is configuring database connections. Visual Studio gives you the opportunity to create mappings between the database connections defined in your project and the databases that are associated with your Azure web site. My `Web.config` file contains only one set of details, and since I only created one Azure database, there is only one entry to pick from the drop-down list. If you have multiple databases in your application, you should take care to ensure that the right Azure database is selected.

Click the `Next` button to preview the effect of your deployment, as shown in Figure 13-11. When you click the `Start Preview` button, Visual Studio walks through the deployment process, but does not actually send the files to the server. If you are upgrading an application that is already deployed, this can be a useful check to ensure that you are only replacing the files that you expect.



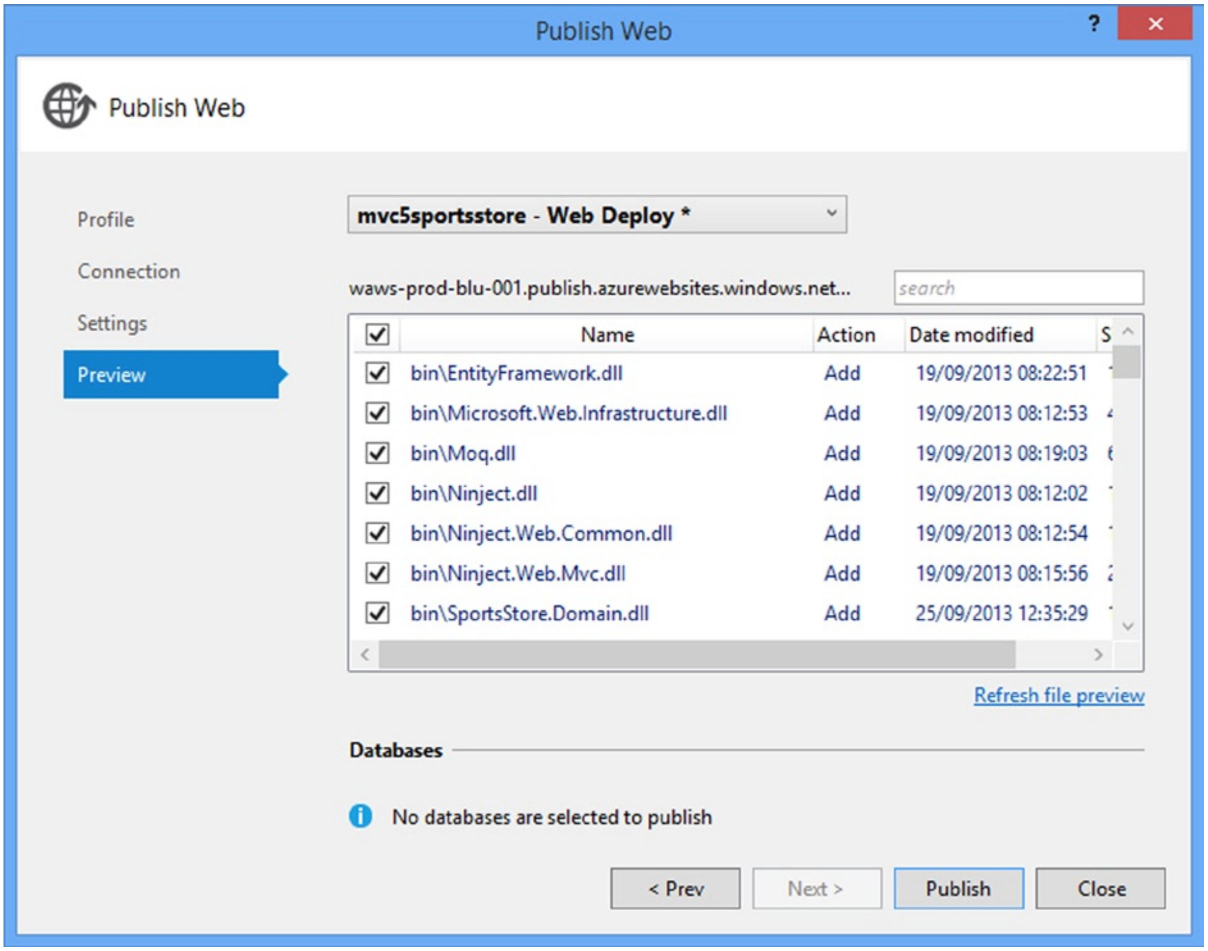
mvc5sportsstore - Web Deploy *

waws-prod-blu-001.publish.azurewebsites.windows.net...

No databases are selected to publish

***Figure 13-11.*** *The Preview section of the Publish Web dialog*

This is the first time that I have deployed this application, so all the files in the project will appear in the preview window, as shown in Figure 13-12. Notice that each file has a check box next to it. You can prevent individual files from being deployed, although you should be careful when doing this. I am pretty conservative in this regard and I would rather deploy files that I do not need rather than forget to deploy one that I do.



waws-prod-blu-001.publish.azurewebsites.windows.net...

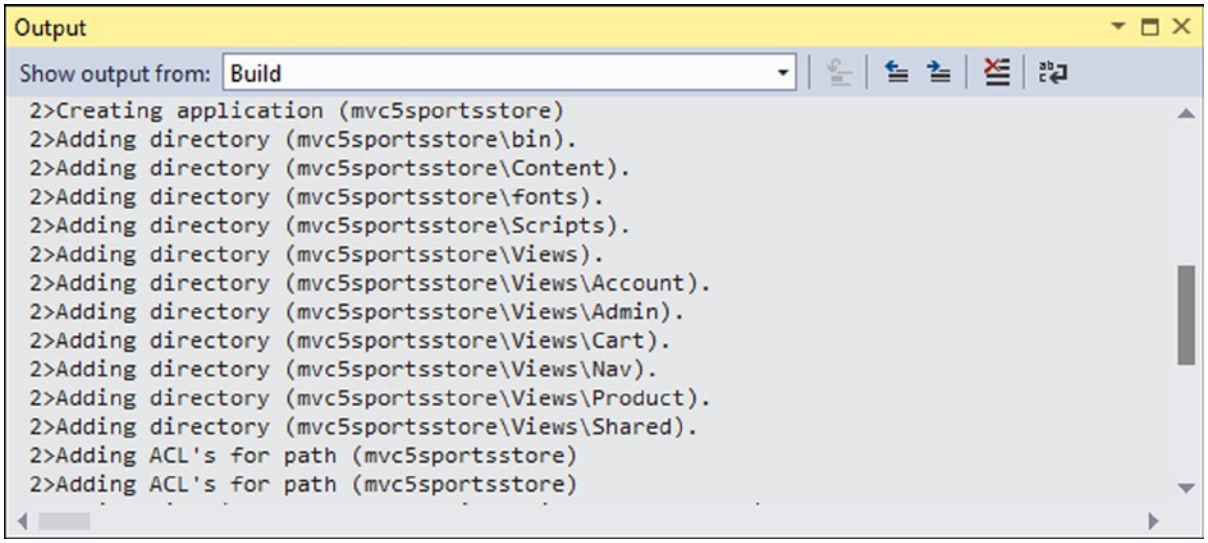
| ✓ | Name | Action | Date modified | S |
|---|---|---|---|---|
| ✓ | bin\EntityFramework.dll | Add | 19/09/2013 08:22:51 | |
| ✓ | bin\Microsoft.Web.Infrastructure.dll | Add | 19/09/2013 08:12:53 | |
| ✓ | bin\Moq.dll | Add | 19/09/2013 08:19:03 | |
| ✓ | bin\Ninject.dll | Add | 19/09/2013 08:12:02 | |
| ✓ | bin\Ninject.Web.Common.dll | Add | 19/09/2013 08:12:54 | |
| ✓ | bin\Ninject.Web.Mvc.dll | Add | 19/09/2013 08:15:56 | |
| ✓ | bin\SportsStore.Domain.dll | Add | 25/09/2013 12:35:29 | |

Databases

No databases are selected to publish

***Figure 13-12.*** *Previewing the deployment changes*

Click the `Publish` button to deploy your application to the Azure platform. The `Publish Web` dialog window will close and you will be able to see details of the deployment progress in the Visual Studio `Output` window, as shown in Figure 13-13.



```
2>Creating application (mvc5sportsstore)
2>Adding directory (mvc5sportsstore\bin).
2>Adding directory (mvc5sportsstore\Content).
2>Adding directory (mvc5sportsstore\fonts).
2>Adding directory (mvc5sportsstore\Scripts).
2>Adding directory (mvc5sportsstore\Views).
2>Adding directory (mvc5sportsstore\Views\Account).
2>Adding directory (mvc5sportsstore\Views\Admin).
2>Adding directory (mvc5sportsstore\Views\Cart).
2>Adding directory (mvc5sportsstore\Views\Nav).
2>Adding directory (mvc5sportsstore\Views\Product).
2>Adding directory (mvc5sportsstore\Views\Shared).
2>Adding ACL's for path (mvc5sportsstore)
2>Adding ACL's for path (mvc5sportsstore)
```

***Figure 13-13.*** *Deploying an application to the Azure platform*

It can take a few minutes to deploy an application, but then the process is complete. Visual Studio will open a browser window that navigates to the URL of your Azure web site. For me, this URL is `http://mvc5sportsstore.azurewebsites.net`, as shown in Figure 13-14.

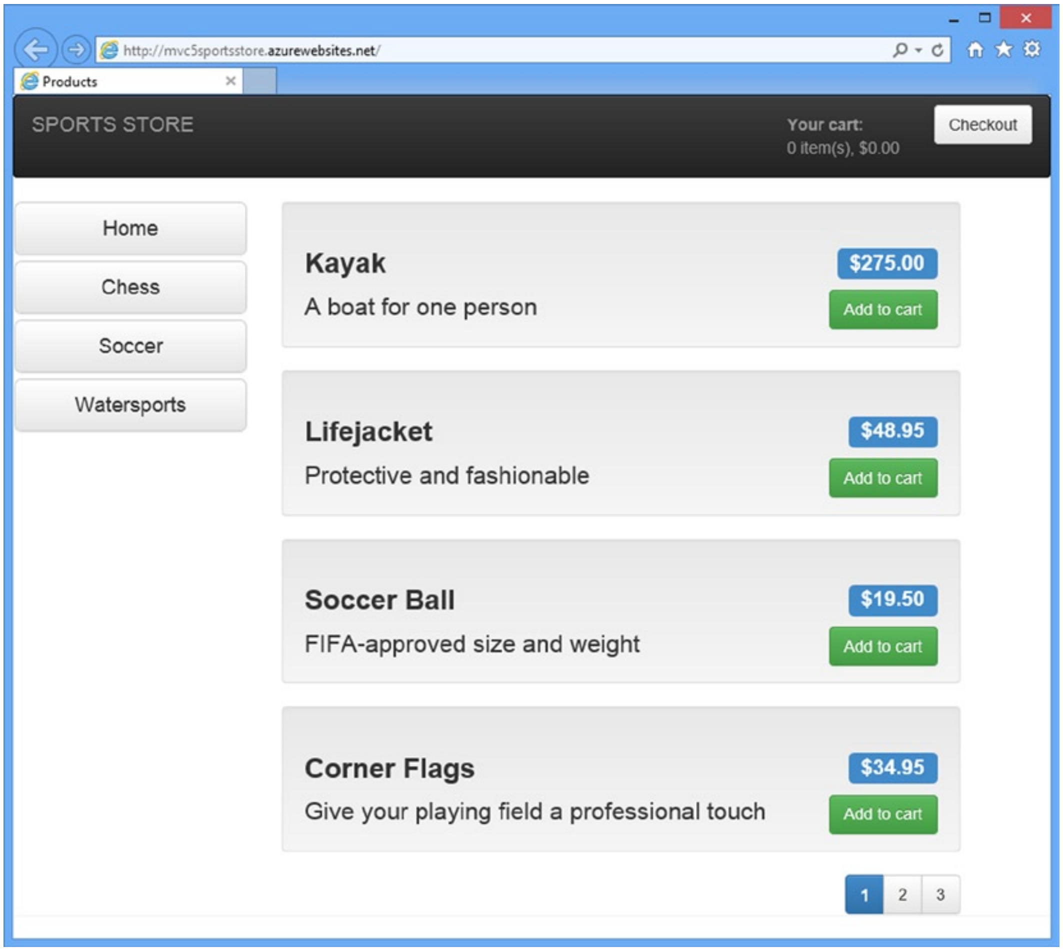


***Figure 13-14.*** *The SportsStore application running on the Windows Azure platform*

## Summary

In this chapter, I have shown you how to deploy an MVC Framework to the Windows Azure platform. There are many different ways to deploy applications and many different platforms that you can target, but the process I have shown you in this chapter is representative of what you can expect, even if you don't use Azure.

And that's the end of the SportsStore application and this part of the book. In Part 2, I begin the process of digging into the detail and showing you how the features I used to create the application work in depth.