

Table of Contents

Preface	xxv
Chapter 1: Hello, C#! Welcome, .NET!	1
Setting up your development environment	2
Choosing the appropriate tool and application type for learning	3
Pros and cons of the .NET Interactive Notebooks extension	3
Using Visual Studio Code for cross-platform development	4
Using GitHub Codespaces for development in the cloud	4
Using Visual Studio for Mac for general development	4
Using Visual Studio for Windows for general development	5
What I used	5
Deploying cross-platform	6
Downloading and installing Visual Studio 2022 for Windows	6
Microsoft Visual Studio for Windows keyboard shortcuts	7
Downloading and installing Visual Studio Code	7
Installing other extensions	8
Understanding Microsoft Visual Studio Code versions	9
Microsoft Visual Studio Code keyboard shortcuts	9
Understanding .NET	10
Understanding .NET Framework	10
Understanding the Mono, Xamarin, and Unity projects	10
Understanding .NET Core	11
Understanding the journey to one .NET	11
Understanding .NET support	12
Understanding .NET Runtime and .NET SDK versions	13
Removing old versions of .NET	14
What is different about modern .NET?	14
Windows development	14
Web development	15
Database development	15
Themes of modern .NET	15
Understanding .NET Standard	15
.NET platforms and tools used by the book editions	16
Understanding intermediate language	17
Comparing .NET technologies	17
Building console apps using Visual Studio 2022	18
Managing multiple projects using Visual Studio 2022	18
Writing code using Visual Studio 2022	18
Compiling and running code using Visual Studio	20

Understanding the compiler-generated folders and files	21
Writing top-level programs	21
Adding a second project using Visual Studio 2022	22
Implicitly imported namespaces	22
Building console apps using Visual Studio Code	24
Managing multiple projects using Visual Studio Code	24
Writing code using Visual Studio Code	24
Compiling and running code using the dotnet CLI	27
Adding a second project using Visual Studio Code	27
Managing multiple files using Visual Studio Code	29
Exploring code using .NET Interactive Notebooks	29
Creating a notebook	30
Writing and running code in a notebook	31
Saving a notebook	32
Adding Markdown and special commands to a notebook	32
Executing code in multiple cells	33
Using .NET Interactive Notebooks for the code in this book	34
Reviewing the folders and files for projects	34
Understanding the common folders and files	35
Understanding the solution code on GitHub	36
Making good use of the GitHub repository for this book	36
Raising issues with the book	36
Giving me feedback	37
Downloading solution code from the GitHub repository	37
Using Git with Visual Studio Code and the command line	38
Cloning the book solution code repository	38
Looking for help	39
Reading Microsoft documentation	39
Getting help for the dotnet tool	39
Getting definitions of types and their members	40
Looking for answers on Stack Overflow	42
Searching for answers using Google	43
Subscribing to the official .NET blog	43
Watching Scott Hanselman's videos	43
Practicing and exploring	43
Exercise 1.1 – Test your knowledge	43
Exercise 1.2 – Practice C# anywhere	44
Exercise 1.3 – Explore topics	44
Summary	45
Chapter 2: Speaking C#	47
Introducing the C# language	47
Understanding language versions and features	47
C# 1.0	48
C# 2.0	48
C# 3.0	48
C# 4.0	48

C# 5.0	49
C# 6.0	49
C# 7.0	49
C# 7.1	49
C# 7.2	50
C# 7.3	50
C# 8	50
C# 9	50
C# 10	50
Understanding C# standards	51
Discovering your C# compiler versions	51
How to output the SDK version	52
Enabling a specific language version compiler	52
Understanding C# grammar and vocabulary	53
Showing the compiler version	53
Understanding C# grammar	55
Statements	55
Comments	55
Blocks	56
Examples of statements and blocks	56
Understanding C# vocabulary	57
Comparing programming languages to human languages	57
Changing the color scheme for C# syntax	57
Help for writing correct code	58
Importing namespaces	59
Implicitly and globally importing namespaces	59
Verbs are methods	62
Nouns are types, variables, fields, and properties	62
Revealing the extent of the C# vocabulary	63
Working with variables	65
Naming things and assigning values	66
Literal values	66
Storing text	66
Understanding verbatim strings	67
Storing numbers	68
Storing whole numbers	68
Exploring whole numbers	69
Storing real numbers	70
Writing code to explore number sizes	70
Comparing double and decimal types	71
Storing Booleans	73
Storing any type of object	73
Storing dynamic types	74
Declaring local variables	76
Specifying the type of a local variable	76
Inferring the type of a local variable	76
Using target-typed new to instantiate objects	78
Getting and setting the default values for types	78

Storing multiple values in an array	79
Exploring more about console applications	80
Displaying output to the user	81
Formatting using numbered positional arguments	81
Formatting using interpolated strings	82
Understanding format strings	82
Getting text input from the user	84
Simplifying the usage of the console	84
Getting key input from the user	85
Passing arguments to a console app	86
Setting options with arguments	88
Handling platforms that do not support an API	90
Practicing and exploring	91
Exercise 2.1 – Test your knowledge	91
Exercise 2.2 – Test your knowledge of number types	92
Exercise 2.3 – Practice number sizes and ranges	92
Exercise 2.4 – Explore topics	93
Summary	93
Chapter 3: Controlling Flow, Converting Types, and Handling Exceptions	95
Operating on variables	95
Exploring unary operators	96
Exploring binary arithmetic operators	97
Assignment operators	98
Exploring logical operators	98
Exploring conditional logical operators	100
Exploring bitwise and binary shift operators	101
Miscellaneous operators	103
Understanding selection statements	103
Branching with the if statement	104
Why you should always use braces with if statements	105
Pattern matching with the if statement	105
Branching with the switch statement	106
Pattern matching with the switch statement	108
Simplifying switch statements with switch expressions	109
Understanding iteration statements	110
Looping with the while statement	110
Looping with the do statement	111
Looping with the for statement	112
Looping with the foreach statement	112
Understanding how foreach works internally	113
Casting and converting between types	113
Casting numbers implicitly and explicitly	114
Converting with the System.Convert type	115
Rounding numbers	116
Understanding the default rounding rules	116
Taking control of rounding rules	117

Converting from any type to a string	117
Converting from a binary object to a string	118
Parsing from strings to numbers or dates and times	119
Errors using Parse	120
Avoiding exceptions using the TryParse method	120
Handling exceptions	121
Wrapping error-prone code in a try block	121
Catching all exceptions	123
Catching specific exceptions	123
Catching with filters	125
Checking for overflow	125
Throwing overflow exceptions with the checked statement	125
Disabling compiler overflow checks with the unchecked statement	127
Practicing and exploring	128
Exercise 3.1 – Test your knowledge	128
Exercise 3.2 – Explore loops and overflow	129
Exercise 3.3 – Practice loops and operators	129
Exercise 3.4 – Practice exception handling	130
Exercise 3.5 – Test your knowledge of operators	130
Exercise 3.6 – Explore topics	130
Summary	130
Chapter 4: Writing, Debugging, and Testing Functions	131
Writing functions	131
Times table example	132
Writing a times table function	132
Writing a function that returns a value	134
Converting numbers from cardinal to ordinal	136
Calculating factorials with recursion	137
Documenting functions with XML comments	140
Using lambdas in function implementations	141
Debugging during development	144
Creating code with a deliberate bug	144
Setting a breakpoint and start debugging	145
Using Visual Studio 2022	145
Using Visual Studio Code	146
Navigating with the debugging toolbar	148
Debugging windows	149
Stepping through code	150
Customizing breakpoints	151
Logging during development and runtime	153
Understanding logging options	153
Instrumenting with Debug and Trace	154
Writing to the default trace listener	154
Configuring trace listeners	155
Switching trace levels	157
Adding packages to a project in Visual Studio Code	157
Adding packages to a project in Visual Studio 2022	158

Reviewing project packages	158
Unit testing	162
Understanding types of testing	162
Creating a class library that needs testing	162
Writing unit tests	164
Running unit tests using Visual Studio Code	165
Running unit tests using Visual Studio	166
Fix the bug	166
Throwing and catching exceptions in functions	167
Understanding usage errors and execution errors	167
Commonly thrown exceptions in functions	167
Understanding the call stack	168
Where to catch exceptions	171
Rethrowing exceptions	171
Implementing the tester-doer pattern	173
Problems with the tester-doer pattern	173
Practicing and exploring	174
Exercise 4.1 – Test your knowledge	174
Exercise 4.2 – Practice writing functions with debugging and unit testing	174
Exercise 4.3 – Explore topics	175
Summary	175
Chapter 5: Building Your Own Types with Object-Oriented Programming	177
Talking about OOP	177
Building class libraries	178
Creating a class library	178
Defining a class in a namespace	179
Simplifying namespace declarations	180
Understanding members	181
Instantiating a class	181
Referencing an assembly	182
Importing a namespace to use a type	182
Understanding objects	183
Inheriting from System.Object	184
Storing data within fields	184
Defining fields	184
Understanding access modifiers	185
Setting and outputting field values	186
Storing a value using an enum type	187
Storing multiple values using an enum type	188
Storing multiple values using collections	189
Understanding generic collections	190
Making a field static	191
Making a field constant	192
Making a field read-only	193
Initializing fields with constructors	194
Defining multiple constructors	195

Writing and calling methods	195
Returning values from methods	195
Combining multiple returned values using tuples	196
Language support for tuples	197
Naming the fields of a tuple	198
Inferring tuple names	198
Deconstructing tuples	198
Deconstructing types	199
Defining and passing parameters to methods	200
Overloading methods	201
Passing optional and named parameters	201
Naming parameter values when calling methods	203
Controlling how parameters are passed	203
Simplified out parameters	204
Understanding ref returns	205
Splitting classes using partial	205
Controlling access with properties and indexers	206
Defining read-only properties	206
Defining settable properties	207
Requiring properties to be set during instantiation	209
Defining indexers	209
Pattern matching with objects	210
Creating and referencing a .NET 6 class library	210
Defining flight passengers	211
Enhancements to pattern matching in C# 9 or later	212
Working with records	213
Init-only properties	213
Understanding records	214
Positional data members in records	215
Simplifying data members in records	215
Practicing and exploring	216
Exercise 5.1 – Test your knowledge	217
Exercise 5.2 – Explore topics	217
Summary	217
Chapter 6: Implementing Interfaces and Inheriting Classes	219
Setting up a class library and console application	220
More about methods	221
Implementing functionality using methods	221
Implementing functionality using operators	223
Implementing functionality using local functions	224
Raising and handling events	225
Calling methods using delegates	226
Defining and handling delegates	227
Defining and handling events	229
Making types safely reusable with generics	230
Working with non-generic types	230

Working with generic types	231
Implementing interfaces	232
Common interfaces	232
Comparing objects when sorting	233
Comparing objects using a separate class	235
Implicit and explicit interface implementations	236
Defining interfaces with default implementations	237
Managing memory with reference and value types	239
Defining reference and value types	239
How reference and value types are stored in memory	240
Equality of types	241
Defining struct types	242
Working with record struct types	243
Releasing unmanaged resources	244
Ensuring that Dispose is called	246
Working with null values	246
Making a value type nullable	246
Understanding nullable reference types	247
Enabling nullable and non-nullable reference types	248
Declaring non-nullable variables and parameters	248
Checking for null	250
Checking for null in method parameters	251
Inheriting from classes	252
Extending classes to add functionality	252
Hiding members	253
Overriding members	254
Inheriting from abstract classes	255
Preventing inheritance and overriding	256
Understanding polymorphism	257
Casting within inheritance hierarchies	259
Implicit casting	259
Explicit casting	259
Avoiding casting exceptions	260
Inheriting and extending .NET types	261
Inheriting exceptions	261
Extending types when you can't inherit	263
Using static methods to reuse functionality	263
Using extension methods to reuse functionality	264
Using an analyzer to write better code	265
Suppressing warnings	267
Fixing the code	268
Understanding common StyleCop recommendations	270
Practicing and exploring	271
Exercise 6.1 – Test your knowledge	271
Exercise 6.2 – Practice creating an inheritance hierarchy	271
Exercise 6.3 – Explore topics	272

Summary	272
Chapter 7: Packaging and Distributing .NET Types	273
The road to .NET 6	273
.NET Core 1.0	274
.NET Core 1.1	274
.NET Core 2.0	275
.NET Core 2.1	275
.NET Core 2.2	275
.NET Core 3.0	275
.NET Core 3.1	276
.NET 5.0	276
.NET 6.0	276
Improving performance from .NET Core 2.0 to .NET 5	277
Checking your .NET SDKs for updates	277
Understanding .NET components	277
Understanding assemblies, NuGet packages, and namespaces	278
What is a namespace?	278
Understanding dependent assemblies	278
Understanding the Microsoft .NET project SDKs	278
Understanding namespaces and types in assemblies	279
Understanding NuGet packages	280
Understanding frameworks	280
Importing a namespace to use a type	281
Relating C# keywords to .NET types	281
Mapping C# aliases to .NET types	282
Revealing the location of a type	283
Sharing code with legacy platforms using .NET Standard	284
Understanding defaults for class libraries with different SDKs	284
Creating a .NET Standard 2.0 class library	285
Controlling the .NET SDK	286
Publishing your code for deployment	287
Creating a console application to publish	288
Understanding dotnet commands	289
Creating new projects	289
Getting information about .NET and its environment	290
Managing projects	291
Publishing a self-contained app	292
Publishing a single-file app	293
Reducing the size of apps using app trimming	295
Enabling assembly-level trimming	295
Enabling type-level and member-level trimming	295
Decompiling .NET assemblies	296
Decompiling using the ILSpy extension for Visual Studio 2022	296
Decompiling using the ILSpy extension for Visual Studio Code	297
No, you cannot technically prevent decompilation	301
Packaging your libraries for NuGet distribution	302

Referencing a NuGet package	302
Fixing dependencies	303
Packaging a library for NuGet	304
Publishing a package to a public NuGet feed	306
Publishing a package to a private NuGet feed	307
Exploring NuGet packages with a tool	307
Testing your class library package	308
Porting from .NET Framework to modern .NET	309
Could you port?	309
Should you port?	310
Differences between .NET Framework and modern .NET	311
Understanding the .NET Portability Analyzer	311
Understanding the .NET Upgrade Assistant	311
Using non-.NET Standard libraries	312
Working with preview features	313
Requiring preview features	314
Enabling preview features	314
Generic mathematics	315
Practicing and exploring	315
Exercise 7.1 – Test your knowledge	316
Exercise 7.2 – Explore topics	316
Exercise 7.3 – Explore PowerShell	316
Summary	316
Chapter 8: Working with Common .NET Types	317
Working with numbers	318
Working with big integers	318
Working with complex numbers	319
Understanding quaternions	320
Working with text	320
Getting the length of a string	320
Getting the characters of a string	321
Splitting a string	321
Getting part of a string	322
Checking a string for content	323
Joining, formatting, and other string members	323
Building strings efficiently	324
Working with dates and times	325
Specifying date and time values	325
Globalization with dates and times	327
Working with only a date or a time	329
Pattern matching with regular expressions	330
Checking for digits entered as text	330
Regular expression performance improvements	331
Understanding the syntax of a regular expression	332
Examples of regular expressions	332

Splitting a complex comma-separated string	333
Storing multiple objects in collections	334
Common features of all collections	335
Improving performance by ensuring the capacity of a collection	336
Understanding collection choices	337
Lists	337
Dictionaries	338
Stacks	339
Queues	339
Sets	340
Collection methods summary	340
Working with lists	340
Working with dictionaries	342
Working with queues	344
Sorting collections	346
More specialized collections	347
Working with a compact array of bit values	347
Working with efficient lists	347
Using immutable collections	347
Good practice with collections	348
Working with spans, indexes, and ranges	349
Using memory efficiently using spans	349
Identifying positions with the Index type	349
Identifying ranges with the Range type	350
Using indexes, ranges, and spans	350
Working with network resources	351
Working with URIs, DNS, and IP addresses	352
Pinging a server	353
Working with reflection and attributes	354
Versioning of assemblies	355
Reading assembly metadata	355
Creating custom attributes	358
Doing more with reflection	360
Working with images	360
Internationalizing your code	362
Detecting and changing the current culture	363
Practicing and exploring	365
Exercise 8.1 – Test your knowledge	365
Exercise 8.2 – Practice regular expressions	366
Exercise 8.3 – Practice writing extension methods	366
Exercise 8.4 – Explore topics	366
Summary	367
Chapter 9: Working with Files, Streams, and Serialization	369
Managing the filesystem	369
Handling cross-platform environments and filesystems	369
Managing drives	371

Managing directories	372
Managing files	374
Managing paths	375
Getting file information	376
Controlling how you work with files	377
Reading and writing with streams	378
Understanding abstract and concrete streams	378
Understanding storage streams	379
Understanding function streams	379
Understanding stream helpers	379
Writing to text streams	380
Writing to XML streams	381
Disposing of file resources	383
Simplifying disposal by using the using statement	385
Compressing streams	386
Compressing with the Brotli algorithm	388
Encoding and decoding text	390
Encoding strings as byte arrays	391
Encoding and decoding text in files	393
Serializing object graphs	394
Serializing as XML	394
Generating compact XML	397
Deserializing XML files	398
Serializing with JSON	399
High-performance JSON processing	400
Controlling JSON processing	401
New JSON extension methods for working with HTTP responses	404
Migrating from Newtonsoft to new JSON	404
Practicing and exploring	405
Exercise 9.1 – Test your knowledge	405
Exercise 9.2 – Practice serializing as XML	405
Exercise 9.3 – Explore topics	406
Summary	406
Chapter 10: Working with Data Using Entity Framework Core	407
Understanding modern databases	407
Understanding legacy Entity Framework	408
Using the legacy Entity Framework 6.3 or later	408
Understanding Entity Framework Core	408
Creating a console app for working with EF Core	409
Using a sample relational database	409
Using Microsoft SQL Server for Windows	410
Downloading and installing SQL Server	411
Creating the Northwind sample database for SQL Server	412
Managing the Northwind sample database with Server Explorer	413
Using SQLite	414
Setting up SQLite for macOS	414

Setting up SQLite for Windows	414
Setting up SQLite for other OSes	414
Creating the Northwind sample database for SQLite	415
Managing the Northwind sample database with SQLiteStudio	415
Setting up EF Core	417
Choosing an EF Core database provider	417
Connecting to a database	417
Defining the Northwind database context class	418
Defining EF Core models	420
Using EF Core conventions to define the model	421
Using EF Core annotation attributes to define the model	421
Using the EF Core Fluent API to define the model	423
Understanding data seeding with the Fluent API	423
Building an EF Core model for the Northwind tables	423
Defining the Category and Product entity classes	424
Adding tables to the Northwind database context class	426
Setting up the dotnet-ef tool	427
Scaffolding models using an existing database	428
Configuring preconvention models	432
Querying EF Core models	433
Filtering included entities	435
Unicode characters in the Windows console	436
Filtering and sorting products	437
Getting the generated SQL	438
Logging EF Core using a custom logging provider	439
Filtering logs by provider-specific values	442
Logging with query tags	443
Pattern matching with Like	444
Defining global filters	445
Loading patterns with EF Core	446
Eager loading entities	446
Enabling lazy loading	447
Explicit loading entities	448
Manipulating data with EF Core	450
Inserting entities	450
Updating entities	452
Deleting entities	453
Pooling database contexts	454
Working with transactions	454
Controlling transactions using isolation levels	455
Defining an explicit transaction	455
Code First EF Core models	456
Understanding migrations	463
Practicing and exploring	464
Exercise 10.1 – Test your knowledge	464
Exercise 10.2 – Practice exporting data using different serialization formats	464

Exercise 10.3 – Explore topics	464
Exercise 10.4 – Explore NoSQL databases	465
Summary	465
Chapter 11: Querying and Manipulating Data Using LINQ	467
Writing LINQ expressions	467
What makes LINQ?	467
Building LINQ expressions with the Enumerable class	468
Understanding deferred execution	470
Filtering entities with Where	471
Targeting a named method	473
Simplifying the code by removing the explicit delegate instantiation	474
Targeting a lambda expression	474
Sorting entities	475
Sorting by a single property using OrderBy	475
Sorting by a subsequent property using ThenBy	475
Declaring a query using var or a specified type	476
Filtering by type	476
Working with sets and bags using LINQ	478
Using LINQ with EF Core	480
Building an EF Core model	480
Filtering and sorting sequences	483
Projecting sequences into new types	485
Joining and grouping sequences	486
Joining sequences	487
Group-joining sequences	488
Aggregating sequences	490
Sweetening LINQ syntax with syntactic sugar	491
Using multiple threads with parallel LINQ	492
Creating an app that benefits from multiple threads	492
Using Windows	494
Using macOS	494
For all operating systems	494
Creating your own LINQ extension methods	495
Trying the chainable extension method	498
Trying the mode and median methods	498
Working with LINQ to XML	499
Generating XML using LINQ to XML	499
Reading XML using LINQ to XML	500
Practicing and exploring	501
Exercise 11.1 – Test your knowledge	501
Exercise 11.2 – Practice querying with LINQ	502
Exercise 11.3 – Explore topics	503
Summary	503
Chapter 12: Improving Performance and Scalability Using Multitasking	505
Understanding processes, threads, and tasks	505
Monitoring performance and resource usage	506

Evaluating the efficiency of types	506
Monitoring performance and memory using diagnostics	507
Useful members of the Stopwatch and Process types	508
Implementing a Recorder class	508
Measuring the efficiency of processing strings	510
Monitoring performance and memory using Benchmark.NET	512
Running tasks asynchronously	516
Running multiple actions synchronously	516
Running multiple actions asynchronously using tasks	518
Starting tasks	518
Waiting for tasks	519
Using wait methods with tasks	519
Continuing with another task	520
Nested and child tasks	522
Wrapping tasks around other objects	523
Synchronizing access to shared resources	524
Accessing a resource from multiple threads	525
Applying a mutually exclusive lock to a conch	526
Understanding the lock statement	527
Avoiding deadlocks	528
Synchronizing events	529
Making CPU operations atomic	530
Applying other types of synchronization	531
Understanding async and await	532
Improving responsiveness for console apps	532
Improving responsiveness for GUI apps	533
Improving scalability for web applications and web services	537
Common types that support multitasking	537
Using await in catch blocks	537
Working with async streams	538
Practicing and exploring	539
Exercise 12.1 – Test your knowledge	539
Exercise 12.2 – Explore topics	539
Summary	539
Chapter 13: Introducing Practical Applications of C# and .NET	541
Understanding app models for C# and .NET	541
Building websites using ASP.NET Core	542
Building websites using a content management system	542
Building web applications using SPA frameworks	543
Building web and other services	544
Building mobile and desktop apps	545
Alternatives to .NET MAUI	545
Understanding Uno Platform	545
Understanding Avalonia	546
New features in ASP.NET Core	546
ASP.NET Core 1.0	546

ASP.NET Core 1.1	546
ASP.NET Core 2.0	546
ASP.NET Core 2.1	547
ASP.NET Core 2.2	547
ASP.NET Core 3.0	548
ASP.NET Core 3.1	548
Blazor WebAssembly 3.2	548
ASP.NET Core 5.0	548
ASP.NET Core 6.0	548
Building Windows-only desktop apps	549
Understanding legacy Windows application platforms	549
Understanding modern .NET support for legacy Windows platforms	550
Structuring projects	550
Structuring projects in a solution or workspace	551
Using other project templates	552
Installing additional template packs	552
Building an entity data model for the Northwind database	553
Creating a class library for entity models using SQLite	554
Improving the class-to-table mapping	555
Creating a class library for a Northwind database context	559
Creating a class library for entity models using SQL Server	562
Practicing and exploring	565
Exercise 13.1 – Test your knowledge	565
Exercise 13.2 – Explore topics	565
Summary	565
Chapter 14: Building Websites Using ASP.NET Core Razor Pages	567
Understanding web development	567
Understanding HTTP	567
Understanding the components of a URL	568
Assigning port numbers for projects in this book	569
Using Google Chrome to make HTTP requests	569
Understanding client-side web development technologies	572
Understanding ASP.NET Core	572
Classic ASP.NET versus modern ASP.NET Core	573
Creating an empty ASP.NET Core project	574
Testing and securing the website	576
Enabling stronger security and redirect to a secure connection	579
Controlling the hosting environment	580
Separating configuration for services and pipeline	582
Enabling a website to serve static content	584
Creating a folder for static files and a web page	584
Enabling static and default files	585
Exploring ASP.NET Core Razor Pages	586
Enabling Razor Pages	586
Adding code to a Razor Page	587
Using shared layouts with Razor Pages	588

Using code-behind files with Razor Pages	591
Using Entity Framework Core with ASP.NET Core	593
Configure Entity Framework Core as a service	593
Manipulating data using Razor Pages	596
Enabling a model to insert entities	596
Defining a form to insert a new supplier	597
Injecting a dependency service into a Razor Page	597
Using Razor class libraries	598
Creating a Razor class library	598
Disabling compact folders for Visual Studio Code	599
Implementing the employees feature using EF Core	600
Implementing a partial view to show a single employee	602
Using and testing a Razor class library	603
Configuring services and the HTTP request pipeline	604
Understanding endpoint routing	604
Configuring endpoint routing	605
Reviewing the endpoint routing configuration in our project	605
Registering services in the ConfigureServices method	606
Setting up the HTTP request pipeline in the Configure method	608
Summarizing key middleware extension methods	609
Visualizing the HTTP pipeline	610
Implementing an anonymous inline delegate as middleware	610
Practicing and exploring	612
Exercise 14.1 – Test your knowledge	612
Exercise 14.2 – Practice building a data-driven web page	613
Exercise 14.3 – Practice building web pages for console apps	613
Exercise 14.4 – Explore topics	613
Summary	613
Chapter 15: Building Websites Using the Model-View-Controller Pattern	615
Setting up an ASP.NET Core MVC website	615
Creating an ASP.NET Core MVC website	616
Creating the authentication database for SQL Server LocalDB	617
Exploring the default ASP.NET Core MVC website	618
Understanding visitor registration	619
Reviewing an MVC website project structure	620
Reviewing the ASP.NET Core Identity database	622
Exploring an ASP.NET Core MVC website	622
Understanding ASP.NET Core MVC initialization	622
Understanding the default MVC route	625
Understanding controllers and actions	626
Understanding the ControllerBase class	626
Understanding the Controller class	627
Understanding the responsibilities of a controller	628
Understanding the view search path convention	629
Understanding logging	630
Understanding filters	631

Using a filter to secure an action method	631
Enabling role management and creating a role programmatically	632
Using a filter to cache a response	635
Using a filter to define a custom route	636
Understanding entity and view models	637
Understanding views	640
Customizing an ASP.NET Core MVC website	643
Defining a custom style	643
Setting up the category images	643
Understanding Razor syntax	643
Defining a typed view	644
Reviewing the customized home page	647
Passing parameters using a route value	648
Understanding model binders in more detail	650
Disambiguating action methods	652
Passing a route parameter	654
Passing a form parameter	654
Validating the model	654
Understanding view helper methods	657
Querying a database and using display templates	659
Improving scalability using asynchronous tasks	662
Making controller action methods asynchronous	662
Practicing and exploring	663
Exercise 15.1 – Test your knowledge	663
Exercise 15.2 – Practice implementing MVC by implementing a category detail page	664
Exercise 15.3 – Practice improving scalability by understanding and implementing async action methods	664
Exercise 15.4 – Practice unit testing MVC controllers	665
Exercise 15.5 – Explore topics	665
Summary	665
Chapter 16: Building and Consuming Web Services	667
Building web services using ASP.NET Core Web API	667
Understanding web service acronyms	667
Understanding Windows Communication Foundation (WCF)	668
An alternative to WCF	668
Understanding HTTP requests and responses for Web APIs	669
Creating an ASP.NET Core Web API project	671
Reviewing the web service's functionality	674
Creating a web service for the Northwind database	675
Creating data repositories for entities	677
Implementing a Web API controller	681
Understanding action method return types	681
Configuring the customer repository and Web API controller	683
Specifying problem details	687
Controlling XML serialization	688
Documenting and testing web services	688

Testing GET requests using a browser	688
Testing HTTP requests with the REST Client extension	690
Making GET requests using REST Client	690
Making other requests using REST Client	692
Understanding Swagger	693
Testing requests with Swagger UI	694
Enabling HTTP logging	700
Consuming web services using HTTP clients	702
Understanding HttpClient	702
Configuring HTTP clients using HttpClientFactory	702
Getting customers as JSON in the controller	703
Enabling Cross-Origin Resource Sharing	705
Implementing advanced features for web services	707
Implementing a Health Check API	708
Implementing Open API analyzers and conventions	709
Implementing transient fault handling	709
Adding security HTTP headers	710
Building web services using minimal APIs	711
Building a weather service using minimal APIs	712
Testing the minimal weather service	714
Adding weather forecasts to the Northwind website home page	714
Practicing and exploring	716
Exercise 16.1 – Test your knowledge	716
Exercise 16.2 – Practice creating and deleting customers with HttpClient	717
Exercise 16.3 – Explore topics	717
Summary	717
Chapter 17: Building User Interfaces Using Blazor	719
Understanding Blazor	719
JavaScript and friends	720
Silverlight – C# and .NET using a plugin	720
WebAssembly – a target for Blazor	720
Understanding Blazor hosting models	720
Understanding Blazor components	721
What is the difference between Blazor and Razor?	722
Comparing Blazor project templates	723
Reviewing the Blazor Server project template	723
Understanding CSS and JavaScript isolation	729
Understanding Blazor routing to page components	729
How to define a routable page component	729
How to navigate Blazor routes	729
How to pass route parameters	730
Understanding base component classes	730
How to use the navigation link component with routes	732
Running the Blazor Server project template	732
Reviewing the Blazor WebAssembly project template	733
Building components using Blazor Server	737

Defining and testing a simple component	737
Making the component a routable page component	738
Getting entities into a component	739
Abstracting a service for a Blazor component	742
Defining forms using the EditForm component	745
Building and using a customer form component	746
Testing the customer form component	749
Building components using Blazor WebAssembly	750
Configuring the server for Blazor WebAssembly	751
Configuring the client for Blazor WebAssembly	754
Testing the Blazor WebAssembly components and service	757
Improving Blazor WebAssembly apps	758
Enabling Blazor WebAssembly AOT	759
Exploring Progressive Web App support	760
Implementing offline support for PWAs	762
Understanding the browser compatibility analyzer for Blazor WebAssembly	762
Sharing Blazor components in a class library	763
Interop with JavaScript	765
Libraries of Blazor components	767
Practicing and exploring	767
Exercise 17.1 – Test your knowledge	768
Exercise 17.2 – Practice by creating a times table component	768
Exercise 17.3 – Practice by creating a country navigation item	768
Exercise 17.4 – Explore topics	769
Summary	769
Epilogue	771
Next steps on your C# and .NET learning journey	771
Polishing your skills with design guidelines	771
Books to take your learning further	772
.NET MAUI delayed	773
Next edition coming November 2022	773
Good luck!	773
Index	775
<hr/>	
Online Chapters	
<hr/>	
Chapter 18: Building and Consuming Specialized Services (Available Online)	
<hr/>	
Chapter 19: Building Mobile and Desktop Apps Using .NET MAUI (Available Online)	
<hr/>	
Chapter 20: Protecting Your Data and Applications (Available Online)	
<hr/>	
Appendix A: Answers to the Test Your Knowledge Questions (Available Online)	
<hr/>	