

Table of Contents

	Acknowledgments	xvii
	Introduction	xix
Part I	Basics of the Microsoft .NET Framework	
1	The Architecture of the .NET Framework Development Platform	3
	Compiling Source Code into Managed Modules	3
	Combining Managed Modules into Assemblies	8
	Loading the Common Language Runtime	9
	Executing Your Assembly's Code	12
	IL and Verification	20
	The .NET Framework Class Library	22
	The Common Type System	26
	The Common Language Specification	28
	Interoperability with Unmanaged Code	33
2	Building, Packaging, Deploying, and Administering Applications and Types	37
	.NET Framework Deployment Goals	38
	Building Types into a Module	39
	Combining Modules to Form an Assembly	48
	Adding Assemblies to a Project Using the Visual Studio .NET IDE	56
	Using the Assembly Linker	57
	Including Resource Files in the Assembly	59
	Assembly Version Resource Information	59
	Version Numbers	63
	Culture	65
	Simple Application Deployment (Privately Deployed Assemblies)	66
	Simple Administrative Control (Configuration)	68

x Table of Contents

3	Shared Assemblies	73
	Two Kinds of Assemblies, Two Kinds of Deployment	74
	Giving an Assembly a Strong Name	75
	The Global Assembly Cache	82
	The Internal Structure of the GAC	88
	Building an Assembly That References a Strongly Named Assembly	90
	Strongly Named Assemblies Are Tamper-Resistant	92
	Delayed Signing	93
	Privately Deploying Strongly Named Assemblies	97
	Side-by-Side Execution	100
	How the Runtime Resolves Type References	101
	Advanced Administrative Control (Configuration)	104
	Publisher Policy Control	110
	Repairing a Faulty Application	113

Part II Working with Types and the Common Language Runtime

4	Type Fundamentals	119
	All Types Are Derived from <code>System.Object</code>	119
	Visual Basic Standard Modules	121
	Casting Between Types	126
	Casting with the <code>CType</code> Operator	129
	Testing an Object's Type with the <code>TypeOf...Is</code> Expression	131
	Namespaces and Assemblies	133
5	Primitive, Reference, and Value Types	139
	Programming Language Primitive Types	139
	Checked and Unchecked Primitive Type Operations	144
	Reference Types and Value Types	146
	Boxing and Unboxing Value Types	153
6	Common Object Operations	165
	Object Equality and Identity	165
	Implementing <code>Equals</code> for a Reference Type Whose Base Classes Don't Override <code>Object's Equals</code>	166

Implementing Equals for a Reference Type When One or More of Its Base Classes Overrides Object's Equals	168
Implementing Equals for a Value Type	169
Summary of Implementing Equals and the Equality and Inequality Operators	172
Identity	172
Object Hash Codes	173
Object Cloning	176

Part III **Designing Types**

7	Type Members and Their Accessibility	181
	Type Members	181
	Accessibility Modifiers and Predefined Attributes	185
	Type Predefined Attributes	187
	Field Predefined Attributes	187
	Method Predefined Attributes	188
8	Constants and Fields	191
	Constants	191
	When Is a Constant Not Always a Constant?	192
	Fields	194
9	Methods	197
	Instance Constructors	197
	Type Constructors	204
	Passing Parameters by Reference to a Method	208
	Passing a Variable Number of Parameters to a Method	211
	Optional Arguments	214
	Static Variables in a Method	217
	Operator Overload Methods	220
	Operators and Programming Language Interoperability	222
	Conversion Operator Methods	226
	How Virtual Methods Are Called	227
	Virtual Method Versioning	228

xii Table of Contents

10	Properties	235
	Parameterless Properties	235
	Parameterful Properties	240
11	Events	245
	Designing a Type That Exposes an Event	246
	Designing a Type That Listens for an Event	252
	A Simpler Way to Register and Unregister Interest in Events	255
Part IV	Essential Types	
12	Working with Text	261
	Characters	261
	The <code>System.String</code> Type	265
	Constructing Strings	265
	Strings Are Immutable	268
	Comparing Strings	269
	String Interning	274
	String Pooling	278
	Examining a String's Characters	279
	Other String Operations	282
	Dynamically Constructing a String Efficiently	282
	Constructing a <code>StringBuilder</code> Object	283
	<code>StringBuilder</code> 's Members	284
	Obtaining a String Representation for an Object	287
	Specific Formats and Cultures	288
	Formatting Multiple Objects into a Single String	292
	Providing Your Own Custom Formatter	294
	Parsing a String to Obtain an Object	297
	Encodings: Converting Between Characters and Bytes	301
	Encoding/Decoding Streams of Characters and Bytes	309
	Base-64 String Encoding and Decoding	310
13	Enumerated Types and Bit Flags	313
	Enumerated Types	313
	Bit Flags	319

14	Arrays	323
	All Arrays Are Implicitly Derived from <code>System.Array</code>	327
	Casting Arrays	329
	Passing and Returning Arrays	331
	Creating Arrays That Have a Nonzero Lower Bound	332
	Fast Array Access	334
	Redimensioning an Array	338
15	Interfaces	343
	Interfaces and Inheritance	343
	Designing an Application That Supports Plug-In Components	350
	Changing Fields in a Boxed Value Type Using Interfaces	351
	Implementing Multiple Interfaces That Have the Same Method Name and Signature	355
	Improving Type Safety and Reducing Boxing	358
16	Custom Attributes	363
	Using Custom Attributes	363
	Defining Your Own Attribute	367
	Attribute Constructor and Field/Property Data Types	371
	Detecting the Use of a Custom Attribute	372
	Matching Two Attribute Instances Against Each Other	377
	Pseudo-Custom Attributes	381
17	Delegates	383
	A First Look at Delegates	383
	Using Delegates to Call Back Static Methods	386
	Using Delegates to Call Back Instance Methods	388
	Demystifying Delegates	389
	Some Delegate History: <code>System.Delegate</code> and <code>System.MulticastDelegate</code>	394
	Comparing Delegates for Equality	395
	Delegate Chains	396
	Having More Control over Invoking a Delegate Chain	401
	Delegates and Reflection	404

Part V Managing Types

18	Exceptions	411
	The Mechanics of Exception Handling	412
	The <code>Try</code> Block	414
	The <code>Catch</code> Block	414
	The <code>Finally</code> Block	416
	What Exactly Is an Exception?	417
	The <code>System.Exception</code> Class	422
	FCL-Defined Exception Classes	424
	Defining Your Own Exception Class	427
	How to Use Exceptions Properly	432
	You Can't Have Too Many <code>Finally</code> Blocks	432
	Don't Catch Everything	433
	Gracefully Recovering from an Exception	435
	Backing Out of a Partially Completed Operation When an Unrecoverable Exception Occurs	436
	Hiding an Implementation Detail	437
	What's Wrong with the FCL	440
	Performance Considerations	442
	Catch Filters	445
	What Happened to the <code>OnError</code> Statement?	449
	Unhandled Exceptions	450
	Controlling What the CLR Does When an Unhandled Exception Occurs	456
	Unhandled Exceptions and Windows Forms	457
	Unhandled Exceptions and ASP.NET Web Forms	459
	Unhandled Exceptions and ASP.NET XML Web Services	460
	Exception Stack Traces	460
	Remoting Stack Traces	463
	Debugging Exceptions	464
	Telling Visual Studio What Kind of Code to Debug	468

19	Automatic Memory Management (Garbage Collection)	471
	Understanding the Basics of Working in a Garbage-Collected Platform	471
	The Garbage Collection Algorithm	475
	Finalization	480
	What Causes <code>Finalize</code> Methods to Get Called	485
	Finalization Internals	487
	The Dispose Pattern: Forcing an Object to Clean Up	491
	Using a Type That Implements the Dispose Pattern	499
	Accounting for Exceptions When Using the Dispose Pattern	504
	An Interesting Dependency Issue	505
	Weak References	506
	Weak Reference Internals	508
	Resurrection	510
	Designing an Object Pool Using Resurrection	512
	Generations	515
	Programmatic Control of the Garbage Collector	521
	Other Garbage Collector Performance Issues	523
	Synchronization-Free Allocations	525
	Scalable Parallel Collections	525
	Concurrent Collections	525
	Large Objects	527
	Monitoring Garbage Collections	528
20	CLR Hosting, AppDomains, and Reflection	529
	Metadata: The Cornerstone of the .NET Framework	529
	CLR Hosting	530
	AppDomains	532
	Accessing Objects Across AppDomain Boundaries	536
	AppDomain Events	537
	Applications and How They Host the CLR and Manage AppDomains	538
	“Yukon”	540
	The Gist of Reflection	540
	Reflecting Over an Assembly’s Types	542
	Reflecting Over an AppDomain’s Assemblies	545

xvi Table of Contents

Reflecting Over a Type's Members: Binding	546
Explicitly Loading Assemblies	547
Loading Assemblies as "Data Files"	550
Building a Hierarchy of Exception-Derived Types	551
Explicitly Unloading Assemblies: Unloading an AppDomain	554
Obtaining a Reference to a <code>System.Type</code> Object	556
Reflecting Over a Type's Members	560
Creating an Instance of a Type	563
Calling a Type's Method	565
Bind Once, Invoke Multiple Times	570
Reflecting Over a Type's Interfaces	576
Reflection Performance	578
Index	581