

Inhaltsverzeichnis

Vorwort	45
 Teil I: Grundlagen	
1 Einstieg in Visual Studio 2010	51
1.1 Die Installation von Visual Studio 2010	51
1.1.1 Überblick über die Produktpalette	51
1.1.2 Anforderungen an Hard- und Software	53
1.1.3 Installation der Express Edition	53
1.1.4 Installation von Visual Studio 2010 Ultimate	54
1.2 Unser allererstes VB-Programm	54
1.2.1 Vorbereitungen	55
1.2.2 Programm schreiben	57
1.2.3 Programm kompilieren und testen	57
1.2.4 Einige Erläuterungen zum Quellcode	58
1.2.5 Konsolenanwendungen sind langweilig	59
1.3 Die Windows-Philosophie	59
1.3.1 Mensch-Rechner-Dialog	59
1.3.2 Objekt- und ereignisorientierte Windows-Programmierung	60
1.3.3 Windows-Programmierung unter Visual Studio 2010	61
1.4 Die Entwicklungsumgebung von Visual Studio 2010	62
1.4.1 Der Startdialog	63
1.4.2 Die wichtigsten Fenster	63
1.5 Microsofts .NET-Technologie	66
1.5.1 Zur Geschichte von .NET	67
1.5.2 .NET-Features und Begriffe	69
1.6 Wichtige Neuigkeiten in Visual Studio 2010	76
1.6.1 Die verschiedenen Pakete	76
1.6.2 Die neue Entwicklungsumgebung	76
1.6.3 Neuheiten im .NET-Framework 4.0	77
1.6.4 VB 2010 – Sprache und Compiler	78

1.7	Praxisbeispiele	80
1.7.1	Windows-Anwendung für Einsteiger	80
1.7.2	Windows-Anwendung für fortgeschrittene Einsteiger	84
2	Grundlagen von Visual Basic	93
2.1	Grundbegriffe	93
2.1.1	Anweisungen	93
2.1.2	Bezeichner	94
2.1.3	Kommentare	95
2.1.4	Zeilenumbruch	96
2.2	Datentypen, Variablen und Konstanten	98
2.2.1	Fundamentale Typen	98
2.2.2	Namensgebung von Variablen	99
2.2.3	Deklaration von Variablen	99
2.2.4	Typinferenz	102
2.2.5	Konstanten deklarieren	103
2.2.6	Gültigkeitsbereiche und Lebensdauer von Deklarationen	103
2.2.7	Lokale Variablen mit Dim	103
2.2.8	Lokale Variablen mit Static	104
2.2.9	Private globale Variablen	105
2.2.10	Public Variablen	105
2.3	Wichtige Datentypen im Überblick	105
2.3.1	Byte, Short, Integer, Long	105
2.3.2	Single, Double und Decimal	106
2.3.3	Char und String	106
2.3.4	Date	107
2.3.5	Boolean	108
2.3.6	Object	108
2.4	Konvertieren von Datentypen	109
2.4.1	Implizite und explizite Konvertierung	109
2.4.2	Welcher Datentyp passt in welchen?	110
2.4.3	Konvertierungsfunktionen	111
2.4.4	CType-Funktion	112
2.4.5	Convert-Klasse	112
2.4.6	Konvertieren von Strings	113
2.4.7	Boxing und Unboxing	115
2.4.8	TryCast-Operator	116
2.4.9	Nullable Types	117

2.5	Operatoren	117
2.5.1	Arithmetische Operatoren	118
2.5.2	Zuweisungsoperatoren	118
2.5.3	Logische Operatoren	119
2.5.4	Vergleichsoperatoren	120
2.5.5	Rangfolge der Operatoren	121
2.6	Kontrollstrukturen	122
2.6.1	Verzweigungsbefehle	122
2.6.2	Schleifenanweisungen	124
2.7	Benutzerdefinierte Datentypen	126
2.7.1	Aufzählungstypen	126
2.7.2	Strukturen	127
2.8	Nutzerdefinierte Funktionen/Prozeduren	129
2.8.1	Deklaration und Syntax	130
2.8.2	Parameterübergabe allgemein	131
2.8.3	Übergabe mit ByVal und ByRef	132
2.8.4	Optionale Parameter	133
2.8.5	Überladene Funktionen/Prozeduren	134
2.9	Praxisbeispiele	135
2.9.1	Vom PAP zum Konsolen-Programm	135
2.9.2	Vom Konsolen- zum Windows-Programm	137
2.9.3	Schleifenanweisungen kennen lernen	139
2.9.4	Methoden überladen	142
3	Objektorientiertes Programmieren	145
3.1	Strukturierte versus objektorientierte Programmierung	145
3.1.1	Was bedeutet strukturiert?	145
3.1.2	Was heißt objektorientiert?	146
3.2	Grundbegriffe der OOP	147
3.2.1	Objekt, Klasse, Instanz	147
3.2.2	Kapselung und Wiederverwendbarkeit	148
3.2.3	Vererbung und Polymorphie	148
3.2.4	Sichtbarkeit von Klassen und ihren Mitgliedern	149
3.2.5	Allgemeiner Aufbau einer Klasse	150
3.3	Ein Objekt erzeugen	151
3.3.1	Referenzieren und Instanzieren	152
3.3.2	Klassische Initialisierung	153
3.3.3	Objekt-Initialisierer	153

3.3.4	Arbeiten mit dem Objekt	153
3.3.5	Zerstören des Objekts	154
3.4	OOP-Einführungsbeispiel	154
3.4.1	Vorbereitungen	154
3.4.2	Klasse definieren	155
3.4.3	Objekt erzeugen und initialisieren	156
3.4.4	Objekt verwenden	156
3.4.5	Intellisense – die hilfreiche Fee	157
3.4.6	Objekt testen	157
3.4.7	Warum unsere Klasse noch nicht optimal ist	158
3.5	Eigenschaften	158
3.5.1	Eigenschaften kapseln	158
3.5.2	Eigenschaften mit Zugriffsmethoden kapseln	161
3.5.3	Lese-/Schreibschutz für Eigenschaften	163
3.5.4	Statische Eigenschaften	163
3.5.5	Selbst implementierende Eigenschaften	164
3.6	Methoden	165
3.6.1	Öffentliche und private Methoden	165
3.6.2	Überladene Methoden	166
3.6.3	Statische Methoden	167
3.7	Ereignisse	168
3.7.1	Ereignisse deklarieren	169
3.7.2	Ereignis auslösen	169
3.7.3	Ereignis auswerten	170
3.7.4	Benutzerdefinierte Ereignisse (Custom Events)	171
3.8	Arbeiten mit Konstruktor und Destruktor	174
3.8.1	Der Konstruktor erzeugt das Objekt	174
3.8.2	Bequemer geht's mit einem Objekt-Initialisierer	176
3.8.3	Destruktor und Garbage Collector räumen auf	177
3.8.4	Mit Using den Lebenszyklus des Objekts kapseln	180
3.9	Vererbung und Polymorphie	180
3.9.1	Vererbungsbeziehungen im Klassendiagramm	180
3.9.2	Überschreiben von Methoden (Method-Overriding)	182
3.9.3	Klassen implementieren	182
3.9.4	Objekte implementieren	187
3.9.5	Polymorphe Methoden	188
3.9.6	Allgemeine Hinweise und Regeln zur Vererbung	191

3.10	Besondere Klassen und Features	192
3.10.1	Abstrakte Klassen	192
3.10.2	Abstrakte Methoden	193
3.10.3	Versiegelte Klassen	193
3.10.4	Partielle Klassen	194
3.10.5	Die Basisklasse System.Object	196
3.10.6	Property-Accessors	197
3.11	Schnittstellen (Interfaces)	197
3.11.1	Definition einer Schnittstelle	197
3.11.2	Implementieren einer Schnittstelle	198
3.11.3	Abfragen, ob Schnittstelle vorhanden ist	199
3.11.4	Mehrere Schnittstellen implementieren	199
3.11.5	Schnittstellenprogrammierung ist ein weites Feld	199
3.12	Praxisbeispiele	200
3.12.1	Eigenschaften sinnvoll kapseln	200
3.12.2	Eine statische Klasse anwenden	203
3.12.3	Vom fetten zum dünnen Client	205
3.12.4	Schnittstellenvererbung verstehen	215
4	Arrays, Strings und Funktionen	221
4.1	Datenfelder (Arrays)	221
4.1.1	Ein Array deklarieren	221
4.1.2	Zugriff auf Array-Elemente	222
4.1.3	Oberen Index ermitteln	222
4.1.4	Explizite Arraygrenzen	222
4.1.5	Arrays erzeugen und initialisieren	222
4.1.6	Zugriff mittels Schleife	223
4.1.7	Mehrdimensionale Arrays	224
4.1.8	Dynamische Arrays	225
4.1.9	Zuweisen von Arrays	226
4.1.10	Arrays aus Strukturvariablen	227
4.1.11	Löschen von Arrays	228
4.1.12	Eigenschaften und Methoden von Arrays	228
4.1.13	Übergabe von Arrays	231
4.2	Zeichenkettenverarbeitung	232
4.2.1	Strings zuweisen	232
4.2.2	StringBuilder	232
4.2.3	Eigenschaften und Methoden eines Strings	233

4.2.4	Kopieren eines Strings in ein Char-Array	235
4.2.5	Wichtige (statische) Methoden der String-Klasse	236
4.3	Zahlen formatieren	238
4.3.1	Die ToString-Methode	238
4.3.2	Die Format-Methode	240
4.4	Vordefinierten Funktionen	241
4.4.1	Mathematik	241
4.4.2	Datums- und Zeitfunktionen	244
4.5	Praxisbeispiele	246
4.5.1	Zeichenketten verarbeiten	246
4.5.2	Methodenaufrufe mit Array-Parametern	249
4.5.3	Zeichenketten mittels StringBuilder addieren	252
5	Weitere wichtige Sprachfeatures	257
5.1	Namespaces (Namensräume)	257
5.1.1	Ein kleiner Überblick	257
5.1.2	Die Imports-Anweisung	259
5.1.3	Namespace-Alias	259
5.1.4	Namespaces in Projekteigenschaften	260
5.1.5	Namespace Alias Qualifizierer	261
5.1.6	Eigene Namespaces einrichten	261
5.2	Überladen von Operatoren	262
5.2.1	Syntaxregeln	262
5.2.2	Praktische Anwendung	263
5.2.3	Konvertierungsoperatoren überladen	264
5.3	Auflistungen (Collections)	265
5.3.1	Beziehungen zwischen den Schnittstellen	265
5.3.2	IEnumerable	266
5.3.3	ICollection	266
5.3.4	ICollection	267
5.3.5	Die ArrayList-Collection	267
5.3.6	Die Hashtable	269
5.4	Generische Datentypen	269
5.4.1	Wie es früher einmal war	270
5.4.2	Typsicherheit durch Generics	271
5.4.3	List-Collection ersetzt ArrayList	273
5.4.4	Über die Vorzüge generischer Collections	274
5.4.5	Typbeschränkungen durch Constraints	275

5.4.6	Collection-Initialisierer	275
5.4.7	Generische Methoden	276
5.5	Was sind Delegates?	276
5.5.1	Delegates sind Methodenzeiger	277
5.5.2	Delegate-Typ definieren	277
5.5.3	Delegate-Objekt erzeugen	278
5.5.4	Delegates vereinfacht instanziiieren	279
5.5.5	Relaxed Delegates	279
5.5.6	Anonyme Methoden	280
5.5.7	Lambda-Ausdrücke	281
5.5.8	Lambda-Ausdrücke in der Task Parallel Library	282
5.6	Dynamische Programmierung	283
5.6.1	Wozu dynamische Programmierung?	284
5.6.2	Das Prinzip der dynamischen Programmierung	284
5.6.3	Kovarianz und Kontravarianz	288
5.7	Weitere Datentypen	289
5.7.1	BigInteger	289
5.7.2	Complex	291
5.7.3	Tuple(Of T)	291
5.7.4	SortedSet(Of T)	292
5.8	Praxisbeispiele	293
5.8.1	ArrayList versus generische List	293
5.8.2	Delegates und Lambda Expressions	297
5.8.3	Mit einem dynamischen Objekt eine Datei durchsuchen	299
6	Einführung in LINQ	305
6.1	LINQ-Grundlagen	305
6.1.1	Die LINQ-Architektur	305
6.1.2	LINQ-Implementierungen	306
6.1.3	Anonyme Typen	306
6.1.4	Erweiterungsmethoden	308
6.2	Abfragen mit LINQ to Objects	309
6.2.1	Grundlegendes zur LINQ-Syntax	309
6.2.2	Zwei alternative Schreibweisen von LINQ Abfragen	310
6.2.3	Übersicht der wichtigsten Abfrage-Operatoren	312
6.3	LINQ-Abfragen im Detail	313
6.3.1	Die Projektionsoperatoren Select und SelectMany	314
6.3.2	Der Restriktionsoperator Where	315

6.3.3	Die Sortierungsoperatoren OrderBy und ThenBy	316
6.3.4	Der Gruppierungsoperator GroupBy	317
6.3.5	Verknüpfen mit Join	319
6.3.6	Aggregat-Operatoren	320
6.3.7	Verzögertes Ausführen von LINQ-Abfragen	321
6.3.8	Konvertierungsmethoden	322
6.3.9	Abfragen mit PLINQ	323
6.4	Praxisbeispiele	326
6.4.1	Die Syntax von LINQ-Abfragen verstehen	326
6.4.2	Aggregat-Abfragen mit LINQ	329

Teil II: Technologien

7	Zugriff auf das Dateisystem	335
7.1	Grundlagen	335
7.1.1	Klassen für Verzeichnis- und Dateioperationen	336
7.1.2	Statische versus Instanzen-Klasse	336
7.2	Operationen auf Verzeichnisebene	337
7.2.1	Verzeichnisse erzeugen und löschen	337
7.2.2	Verzeichnisse verschieben und umbenennen	338
7.2.3	Aktuelles Verzeichnis bestimmen	338
7.2.4	Unterverzeichnisse ermitteln	339
7.2.5	Alle Laufwerke ermitteln	339
7.2.6	Alle im Verzeichnis enthaltene Dateien ermitteln	340
7.2.7	Dateien kopieren und verschieben	341
7.2.8	Dateien umbenennen	341
7.2.9	Dateiattribute feststellen	342
7.2.10	Die FileAttribute-Enumeration	342
7.3	Mehr zur FileInfo-Klasse	343
7.3.1	Weitere wichtige Eigenschaften	343
7.3.2	GetFileSystemInfos-Methode	344
7.4	Zugriffsberechtigungen	344
7.4.1	ACL und ACE	344
7.4.2	SetAccessControl-Methode	345
7.4.3	Zugriffsrechte anzeigen	345

7.5	Weitere wichtige Klassen	346
7.5.1	Die Path-Klasse	346
7.5.2	Die Klasse <i>FileSystemWatcher</i>	347
7.6	Dateidialoge	349
7.6.1	Anzeige und Auswertung	349
7.6.2	Wichtige Eigenschaften	350
7.6.3	Dateifilter	350
7.7	Praxisbeispiele	351
7.7.1	Infos über Verzeichnisse und Dateien gewinnen	351
7.7.2	Die Verzeichnisstruktur in eine <i>TreeView</i> einlesen	354
8	Dateien lesen und schreiben	357
8.1	Grundprinzip der Datenpersistenz	357
8.1.1	Dateien und Streams	357
8.1.2	Die wichtigsten Klassen	358
8.1.3	Erzeugen eines Streams	359
8.2	Dateiparameter	359
8.2.1	<i>FileAccess</i>	359
8.2.2	<i>FileMode</i>	359
8.2.3	<i>FileShare</i>	360
8.3	Textdateien	360
8.3.1	Eine Textdatei beschreiben bzw. neu anlegen	360
8.3.2	Eine Textdatei lesen	362
8.4	Binärdateien	363
8.4.1	Lese-/Schreibzugriff	363
8.4.2	Die Methoden <i>ReadAllBytes</i> und <i>WriteAllBytes</i>	364
8.4.3	Varianten zum Erzeugen von <i>BinaryReader</i> / <i>BinaryWriter</i>	364
8.5	Sequenzielle Dateien	365
8.5.1	Lesen und schreiben von strukturierten Daten	365
8.5.2	Serialisieren von Objekten	366
8.6	Dateien verschlüsseln und komprimieren	367
8.6.1	Das Methodenpärchen <i>Encrypt</i> / <i>Decrypt</i>	367
8.6.2	Verschlüsseln unter Windows XP/Vista/Windows 7	368
8.6.3	Verschlüsseln mit der <i>CryptoStream</i> -Klasse	369
8.6.4	Dateien komprimieren	369
8.7	Memory Mapped Files	371
8.7.1	Grundprinzip	371

8.7.2	Erzeugen eines MMF	371
8.7.3	Erstellen eines Map View	372
8.8	Praxisbeispiele	373
8.8.1	Auf eine Textdatei zugreifen	373
8.8.2	Persistente Daten im Objektbaum speichern	377
8.8.3	Ein Memory Mapped File (MMF) verwenden	385
9	XML in Theorie und Praxis	389
9.1	XML – etwas Theorie	389
9.1.1	Übersicht	389
9.1.2	Der XML-Grundaufbau	392
9.1.3	Wohlgeformte Dokumente	393
9.1.4	Processing Instructions (PI)	396
9.1.5	Elemente und Attribute	396
9.1.6	Verwendbare Zeichensätze	398
9.2	XSD-Schemas	400
9.2.1	XSD-Schemas und ADO.NET	400
9.2.2	XML-Schemas in Visual Studio analysieren	402
9.2.3	XML-Datei mit XSD-Schema erzeugen	405
9.2.4	XSD-Schema aus einer XML-Datei erzeugen	406
9.3	XML-Integration in Visual Basic	407
9.3.1	XML-Literale	407
9.3.2	Einfaches Navigieren durch späte Bindung	410
9.3.3	Die LINQ to XML-API	412
9.3.4	Neue XML-Dokumente erzeugen	414
9.3.5	Laden und Sichern von XML-Dokumenten	416
9.3.6	Navigieren in XML-Daten	417
9.3.7	Auswählen und Filtern	419
9.3.8	Manipulieren der XML-Daten	420
9.3.9	XML-Dokumente transformieren	422
9.4	Verwendung des DOM unter .NET	424
9.4.1	Übersicht	424
9.4.2	DOM-Integration in Visual Basic	426
9.4.3	Laden von Dokumenten	426
9.4.4	Erzeugen von XML-Dokumenten	427
9.4.5	Auslesen von XML-Dateien	429
9.4.6	Direktzugriff auf einzelne Elemente	430

9.4.7	Einfügen von Informationen	431
9.4.8	Suchen in den Baumzweigen	433
9.5	Weitere Möglichkeiten der XML-Verarbeitung	436
9.5.1	Die relationale Sicht auf XML-Daten mit XmlDataDocument	436
9.5.2	XML-Daten aus Objektstrukturen erzeugen	439
9.5.3	Schnelles Suchen in XML-Daten mit XPathNavigator	443
9.5.4	Schnelles Auslesen von XML-Daten mit dem XmlReader	445
9.5.5	Erzeugen von XML-Daten mit XmlWriter	447
9.5.6	XML transformieren mit XSLT	449
9.6	Praxisbeispiele	451
9.6.1	Mit dem DOM in XML-Dokumenten navigieren	451
9.6.2	XML-Daten in eine TreeView einlesen	455
10	Einführung in ADO.NET	459
10.1	Eine kleine Übersicht	459
10.1.1	Die ADO.NET-Klassenhierarchie	459
10.1.2	Die Klassen der Datenprovider	460
10.1.3	Das Zusammenspiel der ADO.NET-Klassen	463
10.2	Das Connection-Objekt	464
10.2.1	Allgemeiner Aufbau	464
10.2.2	OleDbConnection	464
10.2.3	Schließen einer Verbindung	466
10.2.4	Eigenschaften des Connection-Objekts	466
10.2.5	Methoden des Connection-Objekts	468
10.2.6	DerConnectionStringBuilder	469
10.3	Das Command-Objekt	470
10.3.1	Erzeugen und Anwenden eines Command-Objekts	470
10.3.2	Erzeugen mittels CreateCommand-Methode	471
10.3.3	Eigenschaften des Command-Objekts	471
10.3.4	Methoden des Command-Objekts	473
10.3.5	Freigabe von Connection- und Command-Objekten	474
10.4	Parameter-Objekte	476
10.4.1	Erzeugen und Anwenden eines Parameter-Objekts	476
10.4.2	Eigenschaften des Parameter-Objekts	476
10.5	Das CommandBuilder-Objekt	477
10.5.1	Erzeugen	477
10.5.2	Anwenden	478

10.6	Das DataReader-Objekt	478
10.6.1	DataReader erzeugen	479
10.6.2	Daten lesen	479
10.6.3	Eigenschaften des DataReaders	480
10.6.4	Methoden des DataReaders	480
10.7	Das DataAdapter-Objekt	481
10.7.1	DataAdapter erzeugen	481
10.7.2	Command-Eigenschaften	482
10.7.3	Fill-Methode	483
10.7.4	Update-Methode	484
10.8	Praxisbeispiele	485
10.8.1	Wichtige ADO.NET-Objekte im Einsatz	485
10.8.2	Eine Aktionsabfrage ausführen	487
10.8.3	Die Datenbank aktualisieren	489
10.8.4	DenConnectionString in Konfigurationsdatei ablegen	492
11	Das DataSet	495
11.1	Grundlegende Features des DataSets	495
11.1.1	Die Objekthierarchie	496
11.1.2	Die wichtigsten Klassen	496
11.1.3	Erzeugen eines DataSets	497
11.2	Das DataTable-Objekt	498
11.2.1	DataTable erzeugen	499
11.2.2	Spalten hinzufügen	499
11.2.3	Zeilen zur DataTable hinzufügen	500
11.2.4	Auf den Inhalt einer DataTable zugreifen	501
11.3	Die DataView	503
11.3.1	Erzeugen eines DataView	503
11.3.2	Sortieren und Filtern von Datensätzen	503
11.3.3	Suchen von Datensätzen	504
11.4	Typisierte DataSets	504
11.4.1	Ein typisiertes DataSet erzeugen	504
11.4.2	Das Konzept der Datenquellen	506
11.4.3	Typisierte DataSets und TableAdapter	506
11.5	Die Qual der Wahl	508
11.5.1	DataReader – der schnelle Lesezugriff	508
11.5.2	DataSet – die Datenbank im Hauptspeicher	509
11.5.3	Objektrelationales Mapping – die Zukunft?	510

11.6	Praxisbeispiele	511
11.6.1	Im DataView sortieren und filtern	511
11.6.2	Suche nach Datensätzen	513
11.6.3	Ein DataSet in einen XML-String serialisieren	514
11.6.4	Ein untypisiertes in ein typisiertes DataSet konvertieren	519
11.6.5	Eine LINQ to SQL-Abfrage ausführen	524
12	Asynchrone Programmierung	529
12.1	Übersicht	529
12.1.1	Multitasking versus Multithreading	530
12.1.2	Deadlocks	531
12.1.3	Racing	531
12.2	Programmieren mit Threads	533
12.2.1	Einführungsbeispiel	533
12.2.2	Wichtige Thread-Methoden	534
12.2.3	Wichtige Thread-Eigenschaften	536
12.2.4	Einsatz der ThreadPool-Klasse	537
12.3	Sperrmechanismen	539
12.3.1	Threading ohne SyncLock	539
12.3.2	Threading mit SyncLock	540
12.3.3	Die Monitor-Klasse	543
12.3.4	Mutex	546
12.3.5	Methoden für die parallele Ausführung sperren	547
12.3.6	Semaphore	548
12.4	Interaktion mit der Programmoberfläche	549
12.4.1	Die Werkzeuge	550
12.4.2	Einzelne Steuerelemente mit Invoke aktualisieren	550
12.4.3	Mehrere Steuerelemente aktualisieren	551
12.4.4	Ist ein Invoke-Aufruf nötig?	552
12.4.5	Und was ist mit WPF?	552
12.5	Timer-Threads	554
12.6	Verwendung der BackgroundWorker-Komponente	555
12.7	Asynchrone Programmier-Entwurfsmuster	558
12.7.1	Kurzübersicht	558
12.7.2	Polling	559
12.7.3	Callback verwenden	560
12.7.4	Callback mit Parameterübergabe verwenden	561
12.7.5	Callback mit Zugriff auf die Programm-Oberfläche	562

12.8	Asynchroner Aufruf beliebiger Methoden	563
12.8.1	Die Beispielklasse	563
12.8.2	Asynchroner Aufruf ohne Callback	565
12.8.3	Asynchroner Aufruf mit Callback und Anzeigefunktion	565
12.8.4	Aufruf mit Rückgabewerten (per Eigenschaft)	566
12.8.5	Aufruf mit Rückgabewerten (per EndInvoke)	567
12.9	Praxisbeispiele	568
12.9.1	Spieltrieb & Multithreading erleben	568
12.9.2	Prozess- und Thread-Informationen gewinnen	580
12.9.3	Ein externes Programm starten	584
13	Die Task Parallel Library	587
13.1	Überblick	587
13.1.1	Parallel-Programmierung	587
13.1.2	Möglichkeiten der TPL	589
13.1.3	Der CLR-Threadpool	590
13.2	Parallele Verarbeitung mit Parallel.Invoke	590
13.2.1	Aufrufvarianten	591
13.2.2	Einschränkungen	592
13.3	Verwendung von Parallel.For	593
13.3.1	Abbrechen der Verarbeitung	594
13.3.2	Auswerten des Bearbeitungsstatus	595
13.3.3	Und was ist mit anderen Schrittweiten für den Iterator?	596
13.4	Verarbeiten von Collections mit Parallel.ForEach	596
13.5	Die neue Task-Klasse	597
13.5.1	Einen Task erzeugen	597
13.5.2	Task starten	598
13.5.3	Datenübergabe an den Task	599
13.5.4	Wie warte ich auf das Taskende?	601
13.5.5	Tasks mit Rückgabewerten	602
13.5.6	Die Verarbeitung abbrechen	605
13.5.7	Fehlerbehandlung	609
13.5.8	Weitere Eigenschaften	610
13.6	Zugriff auf das Userinterface	611
13.6.1	Reaktion auf das Task-Ende und Zugriff auf die Oberfläche	611
13.6.2	Zugriff auf das UI aus dem Task heraus	612

13.7	Weitere Datenstrukturen im Überblick	615
13.7.1	Threadsichere Collections	615
13.7.2	Primitive für die Threadsynchronisation	615
13.8	Parallel LINQ (PLINQ)	616
13.9	Die Parallel Diagnostic Tools	616
13.9.1	Fenster für parallele Aufgaben	616
13.9.2	Fenster für parallele Stacks	617
13.9.3	IntelliTrace	618
13.10	Praxisbeispiel: Spieltrieb – die zweite Version	619
13.10.1	Aufgabenstellung	619
13.10.2	Global-Klasse	619
13.10.3	Controller	621
13.10.4	LKWs	622
13.10.5	Schiff-Klasse	623
13.10.6	Oberfläche	625
14	Fehlersuche und Behandlung	627
14.1	Der Debugger	627
14.1.1	Allgemeine Beschreibung	627
14.1.2	Einzel schrittmodus	631
14.1.3	Prozedurschrittmodus	632
14.1.4	Haltepunkte	632
14.1.5	Debugging am Beispiel	632
14.1.6	Das Debug-Objekt	636
14.2	Fehlerbehandlung	637
14.2.1	Anweisungen zur Fehlerbehandlung	637
14.2.2	Try-Catch	638
14.2.3	Try-Finally	642
14.2.4	Das Standardverhalten bei Ausnahmen festlegen	644
14.2.5	Die Exception-Klasse	645
14.2.6	Fehler/Ausnahmen auslösen	645
14.2.7	Eigene Fehlerklassen	646
14.2.8	Exceptionhandling zur Entwurfszeit	648
15	Unit-Tests	649
15.1	Testgetriebene Entwicklung (TDD)	649
15.1.1	Konventionelle Vorgehensweise	649
15.1.2	Testgetriebene Entwicklung	650

15.1.3	Unit-Tests unter Visual Studio	650
15.2	Einfache Tests	651
15.2.1	Beispiel: Test einer Klasse CKugel	651
15.2.2	Eigene Testmethoden hinzufügen	657
15.3	Datengetriebene Tests (DDT)	657
15.3.1	Das DDT-Prinzip	658
15.3.2	Beispiel: Test von Spesenberechnungen	658
15.4	Begriffe und Ergänzungen	662
15.4.1	Behauptungen (Asserts)	662
15.4.2	Der Testkontext	663
15.4.3	Zusätzliche Testattribute	664
15.5	Praktische Einsatzkriterien	665
15.5.1	Nutzen von Unit-Tests	665
15.5.2	Grenzen von Unit-Tests	665
16	OOP-Spezial	667
16.1	Eine kleine Einführung in die UML	667
16.1.1	Use Case-Diagramm	667
16.1.2	Use Case-Dokumentation	669
16.1.3	Objekte identifizieren	670
16.1.4	Statisches Modell	671
16.1.5	Beziehungen zwischen den Klassen	672
16.1.6	Dynamisches Modell	672
16.1.7	Implementierung	673
16.1.8	Test-Client	677
16.2	Der Klassen-Designer	680
16.2.1	Ein neues Klassendiagramm erzeugen	680
16.2.2	Toolbox	682
16.2.3	Enumeration	683
16.2.4	Klasse	684
16.2.5	Struktur	686
16.2.6	Abstrakte Klasse	687
16.2.7	Schnittstelle	689
16.2.8	Delegate	691
16.2.9	Zuordnung	693
16.2.10	Vererbung	694
16.2.11	Diagramme anpassen	694

16.2.12	Objekt-Testcenter	695
16.2.13	Wann lohnt sich der Einsatz des Klassen-Designers?	697
16.3	Praxisbeispiele	698
16.3.1	Implementierung einer Finite State Machine	698
16.3.2	Modellierung des Bestellsystems einer Firma	703
17	Das Microsoft Event Pattern	717
17.1	Was sind Design Pattern und wozu braucht man sie?	717
17.2	Aufbau und Bedeutung des Observer Pattern	718
17.2.1	Subjekt und Observer	718
17.2.2	Sequenzdiagramme	720
17.2.3	Die Registration-Sequenz	720
17.2.4	Die Notification-Sequenz	721
17.2.5	Die Unregistration-Sequenz	721
17.2.6	Was bedeuten diese Sequenzen für unser Geschäftsmodell?	722
17.2.7	Die Rolle des Containers	722
17.3	Implementierung mit Interfaces und Callbacks	723
17.3.1	Übersicht und Klassendiagramm	723
17.3.2	Schnittstellen IObservable und IObservable	725
17.3.3	Abstrakte Klasse Subject	725
17.3.4	Observer1	726
17.3.5	Observer2	727
17.3.6	Model	728
17.3.7	Form1	729
17.3.8	Ein zweites Klassendiagramm	730
17.3.9	Testen	731
17.4	Implementierung mit Delegates und Events	732
17.4.1	Multicast-Events	733
17.4.2	IObservable, IObservable und Subject	733
17.4.3	Observer1 und Observer2	734
17.4.4	Model	734
17.4.5	Form1	734
17.4.6	Test und Vergleich	735
17.4.7	Klassendiagramm	736
17.5	Implementierung des Microsoft Event Pattern	737
17.5.1	Namensgebung für Ereignisse	737
17.5.2	Namensgebung und Signatur der Delegates	737
17.5.3	Hinzufügen einer das Ereignis auslösenden Methode	738

17.5.4	Neue Klasse NumberChangedEventArgs	738
17.5.5	Model	739
17.5.6	Observer1	740
17.5.7	Form1	740
17.6	Test und Vergleich	741
17.7	Klassendiagramm	741
17.8	Schritte zur Implementierung eines Event Pattern	742
17.9	Praxisbeispiel	743
17.9.1	Subjekt und Observer beobachten sich gegenseitig	743
18	Weitere Programmiertechniken	753
18.1	Zugriff auf die Zwischenablage	753
18.1.1	Das Clipboard-Objekt	753
18.1.2	Zwischenablage-Funktionen für Textboxen	755
18.2	Arbeiten mit der Registry	755
18.2.1	Allgemeines	756
18.2.2	Registry-Unterstützung in .NET	757
18.3	.NET-Reflection	759
18.3.1	Übersicht	759
18.3.2	Assembly laden	759
18.3.3	Mittels GetType und Type Informationen sammeln	760
18.3.4	Dynamisches Laden von Assemblies	762
18.4	Das SerialPort-Control	764
18.4.1	Übersicht	765
18.4.2	Einführungsbeispiele	766
18.4.3	Thread-Probleme bei Windows Forms Anwendungen	768
18.4.4	Ein einfaches Terminalprogramm	771
18.5	Praxisbeispiele	775
18.5.1	Zugriff auf die Registry	775
18.5.2	Dateiverknüpfungen erzeugen	777
19	Konsolenanwendungen	781
19.1	Grundaufbau/Konzepte	781
19.1.1	Unser Hauptprogramm – Module1.vb	782
19.1.2	Rückgabe eines Fehlerstatus	783
19.1.3	Parameterübergabe	784
19.1.4	Zugriff auf die Umgebungsvariablen	785

19.2	Die Kommandozentrale: System.Console	786
19.2.1	Eigenschaften	787
19.2.2	Methoden/Ereignisse	787
19.2.3	Textausgaben	788
19.2.4	Farbangaben	789
19.2.5	Tastaturabfragen	790
19.2.6	Arbeiten mit Streamdaten	791
19.3	Praxisbeispiel: Farbige Konsolenanwendung	793
20	Verteilen von Anwendungen	795
20.1	ClickOnce-Deployment	795
20.1.1	Übersicht/Einschränkungen	795
20.1.2	Die Vorgehensweise	796
20.1.3	Ort der Veröffentlichung	797
20.1.4	Anwendungsdateien	797
20.1.5	Erforderliche Komponenten	798
20.1.6	Aktualisierungen	799
20.1.7	Veröffentlichen	799
20.1.8	Verzeichnisstruktur	800
20.1.9	Der Webpublishing-Assistent	802
20.1.10	Neue Versionen erstellen	802
20.2	Setup-Projekte	803
20.2.1	Ein neues Setup-Projekt	803
20.2.2	Dateisystem-Editor	805
20.2.3	Ein erster Test	807
20.2.4	Registrierungs-Editor	807
20.2.5	Dateityp-Editor	808
20.2.6	Benutzeroberflächen-Editor	809
20.2.7	Editor für Startbedingungen	812
20.2.8	Finaler Test	813

Teil III: Konsole/Windows Forms

21	Windows Forms-Anwendungen	817
21.1	Grundaufbau/Konzepte	817
21.1.1	Wo ist das Hauptprogramm?	818
21.1.2	Die Oberflächendefinition – Form1.Designer.vb	823

21.1.3	Die Spielwiese des Programmierers – Form1.vb	824
21.1.4	Die Datei AssemblyInfo.vb	825
21.1.5	Resources.resx/Resources.Designer.vb	826
21.1.6	Settings.settings/Settings.Designer.vb	827
21.1.7	Settings.vb	829
21.2	Ein Blick auf die Application-Klasse	829
21.2.1	Eigenschaften	830
21.2.2	Methoden	831
21.2.3	Ereignisse	832
21.3	Allgemeine Eigenschaften von Komponenten	833
21.3.1	Font	833
21.3.2	Handle	835
21.3.3	Tag	836
21.3.4	Modifiers	836
21.4	Allgemeine Ereignisse von Komponenten	837
21.4.1	Die Eventhandler-Argumente	837
21.4.2	Sender	837
21.4.3	Der Parameter e	839
21.4.4	Mausereignisse	839
21.4.5	KeyPreview	841
21.4.6	Weitere Ereignisse	842
21.4.7	Validitätsprüfungen	843
21.4.8	SendKeys	843
21.5	Allgemeine Methoden von Komponenten	845
22	Windows-Formulare verwenden	847
22.1	Übersicht	847
22.1.1	Wichtige Eigenschaften des Form-Objekts	848
22.1.2	Wichtige Ereignisse des Form-Objekts	850
22.1.3	Wichtige Methoden des Form-Objekts	851
22.2	Praktische Aufgabenstellungen	852
22.2.1	Fenster anzeigen	852
22.2.2	Einen Splash Screen beim Anwendungsstart anzeigen	855
22.2.3	Eine Sicherheitsabfrage vor dem Schließen anzeigen	857
22.2.4	Ein Formular durchsichtig machen	858
22.2.5	Die Tabulatorreihenfolge festlegen	858
22.2.6	Ausrichten und Platzieren von Komponenten im Formular	859

22.2.7	Spezielle Panels für flexibles Layout	862
22.2.8	Menüs erzeugen	863
22.3	MDI-Anwendungen	867
22.3.1	Die "falschen" MDI-Fenster bzw. Verwenden von Parent	867
22.3.2	Die echten MDI-Fenster	868
22.3.3	Die Kindfenster	869
22.3.4	Automatisches Anordnen der Kindfenster	870
22.3.5	Zugriff auf die geöffneten MDI-Kindfenster	872
22.3.6	Zugriff auf das aktive MDI-Kindfenster	872
22.3.7	Kombinieren der Kindfenstermenüs mit dem MDIContainer	872
22.4	Praxisbeispiele	873
22.4.1	Informationsaustausch zwischen Formularen	873
22.4.2	Ereigniskette beim Laden/Entladen eines Formulars	882
23	Komponenten-Übersicht	887
23.1	Allgemeine Hinweise	887
23.1.1	Hinzufügen von Komponenten	887
23.1.2	Komponenten zur Laufzeit per Code erzeugen	888
23.2	Allgemeine Steuerelemente	890
23.2.1	Label	890
23.2.2	LinkLabel	891
23.2.3	Button	893
23.2.4	TextBox	893
23.2.5	MaskedTextBox	896
23.2.6	CheckBox	897
23.2.7	RadioButton	899
23.2.8	ListBox	900
23.2.9	CheckedListBox	901
23.2.10	ComboBox	902
23.2.11	PictureBox	903
23.2.12	DateTimePicker	903
23.2.13	MonthCalendar	904
23.2.14	HScrollBar, VScrollBar	904
23.2.15	TrackBar	905
23.2.16	NumericUpDown	906
23.2.17	DomainUpDown	907
23.2.18	ProgressBar	907
23.2.19	RichTextBox	908

23.2.20	ListView	909
23.2.21	TreeView	915
23.2.22	WebBrowser	920
23.3	Container	921
23.3.1	FlowLayout/TableLayout/SplitContainer	921
23.3.2	Panel	921
23.3.3	GroupBox	922
23.3.4	TabControl	923
23.3.5	ImageList	925
23.4	Menüs & Symbolleisten	926
23.4.1	MenuStrip und ContextMenuStrip	926
23.4.2	ToolStrip	926
23.4.3	StatusStrip	927
23.4.4	ToolStripContainer	927
23.5	Daten	927
23.5.1	DataSet	927
23.5.2	DataGridView/DataGrid	928
23.5.3	BindingNavigator/BindingSource	928
23.5.4	Chart	928
23.6	Komponenten	929
23.6.1	ErrorProvider	930
23.6.2	HelpProvider	930
23.6.3	ToolTip	930
23.6.4	Timer	930
23.6.5	BackgroundWorker	930
23.6.6	SerialPort	931
23.7	Drucken	931
23.7.1	PrintPreviewControl	931
23.7.2	PrintDocument	931
23.8	Dialoge	931
23.8.1	OpenFileDialog/SaveFileDialog/FolderBrowserDialog	931
23.8.2	FontDialog/ColorDialog	931
23.9	WPF-Unterstützung mit dem ElementHost	932
23.10	Praxisbeispiele	932
23.10.1	Mit der CheckBox arbeiten	932
23.10.2	Steuerelemente per Code selbst erzeugen	934
23.10.3	Controls-Auflistung des Formulars im TreeView anzeigen	936
23.10.4	WPF-Komponenten mit dem ElementHost anzeigen	939

24 Einführung Grafikausgabe	945
24.1 Übersicht und erste Schritte	945
24.1.1 GDI+ – Ein erster Blick für Umsteiger	946
24.1.2 Namespaces für die Grafikausgabe	947
24.2 Darstellen von Grafiken	949
24.2.1 Die PictureBox-Komponente	949
24.2.2 Das Image-Objekt	950
24.2.3 Laden von Grafiken zur Laufzeit	951
24.2.4 Sichern von Grafiken	951
24.2.5 Grafikeigenschaften ermitteln	952
24.2.6 Erzeugen von Vorschaugrafiken (Thumbnails)	953
24.2.7 Die Methode RotateFlip	954
24.2.8 Skalieren von Grafiken	955
24.3 Das .NET-Koordinatensystem	956
24.3.1 Globale Koordinaten	957
24.3.2 Seitenkoordinaten (globale Transformation)	958
24.3.3 Gerätekoordinaten (Seitentransformation)	960
24.4 Grundlegende Zeichenfunktionen von GDI+	961
24.4.1 Das zentrale Graphics-Objekt	961
24.4.2 Punkte zeichnen/abfragen	964
24.4.3 Linien	965
24.4.4 Kantenglättung mit Antialiasing	966
24.4.5 PolyLine	967
24.4.6 Rechtecke	967
24.4.7 Polygone	969
24.4.8 Splines	970
24.4.9 Bézierkurven	971
24.4.10 Kreise und Ellipsen	972
24.4.11 Tortenstück (Segment)	972
24.4.12 Bogenstück	974
24.4.13 Wo sind die Rechtecke mit den "runden Ecken"?	975
24.4.14 Textausgabe	976
24.4.15 Ausgabe von Grafiken	980
24.5 Unser Werkzeugkasten	981
24.5.1 Einfache Objekte	981
24.5.2 Vordefinierte Objekte	983
24.5.3 Farben/Transparenz	985
24.5.4 Stifte (Pen)	986

24.5.5	Pinsel (Brush)	989
24.5.6	SolidBrush	990
24.5.7	HatchBrush	990
24.5.8	TextureBrush	991
24.5.9	LinearGradientBrush	992
24.5.10	PathGradientBrush	994
24.5.11	Fonts	995
24.5.12	Path-Objekt	996
24.5.13	Clipping/Region	999
24.6	Standarddialoge	1002
24.6.1	Schriftauswahl	1002
24.6.2	Farbauswahl	1003
24.7	Praxisbeispiele	1005
24.7.1	Ein Graphics-Objekt erzeugen	1005
24.7.2	Zeichenoperationen mit der Maus realisieren	1008
25	Druckausgabe	1013
25.1	Einstieg und Übersicht	1013
25.1.1	Nichts geht über ein Beispiel	1013
25.1.2	Programmiermodell	1015
25.1.3	Kurzübersicht der Objekte	1016
25.2	Auswerten der aktuellen Druckereinstellungen	1016
25.2.1	Die vorhandenen Drucker	1016
25.2.2	Der Standarddrucker	1017
25.2.3	Verfügbare Papierformate/Seitenabmessungen	1017
25.2.4	Der eigentliche Druckbereich	1019
25.2.5	Die Seitenausrichtung ermitteln	1020
25.2.6	Ermitteln der Farbfähigkeit	1020
25.2.7	Die Druckauflösung abfragen	1020
25.2.8	Ist beidseitiger Druck möglich?	1021
25.2.9	Einen "Informationsgerätekontext" erzeugen	1021
25.2.10	Abfragen von Werten während des Drucks	1022
25.3	Festlegen von Druckereinstellungen	1023
25.3.1	Einen Drucker auswählen	1023
25.3.2	Drucken in Millimetern	1023
25.3.3	Festlegen der Seitenränder	1024
25.3.4	Druckjobname	1025
25.3.5	Anzahl der Kopien	1026
25.3.6	Beidseitiger Druck	1026

25.3.7	Seitenzahlen festlegen	1027
25.3.8	Druckqualität verändern	1030
25.3.9	Ausgabemöglichkeiten des Chart-Controls nutzen	1030
25.4	Die Druckdialoge verwenden	1031
25.4.1	PrintDialog	1031
25.4.2	PageSetupDialog	1032
25.4.3	PrintPreviewDialog	1034
25.4.4	Ein eigenes Druckvorschau-Fenster realisieren	1035
25.5	Drucken mit OLE-Automation	1036
25.5.1	Kurzeinstieg in die OLE-Automation	1036
25.5.2	Drucken mit Microsoft Word	1040
25.6	Praxisbeispiele	1042
25.6.1	Den Drucker umfassend konfigurieren	1042
25.6.2	Diagramme mit dem Chart-Control drucken	1052
25.6.3	Drucken mit Word	1054
26	Windows Forms-Datenbindung	1061
26.1	Prinzipielle Möglichkeiten	1061
26.2	Manuelle Datenbindung an einfache Datenfelder	1062
26.2.1	BindingSource erzeugen	1062
26.2.2	Binding-Objekt	1063
26.2.3	DataBindings-Collection	1063
26.3	Manuelle Datenbindung an Listen und Tabelleninhalte	1063
26.3.1	DataGridView	1064
26.3.2	Datenbindung von ComboBox und ListBox	1064
26.4	Entwurfszeit-Datenbindung an typisierte DataSets	1064
26.5	Drag & Drop-Datenbindung	1066
26.6	Navigations- und Bearbeitungsfunktionen realisieren	1066
26.6.1	Navigieren zwischen den Datensätzen	1066
26.6.2	Hinzufügen und Löschen	1066
26.6.3	Aktualisieren und Abbrechen	1067
26.6.4	Verwendung des BindingNavigators	1067
26.7	Die Anzeigedaten formatieren	1068
26.8	Praxisbeispiele	1068
26.8.1	Einrichten und Verwenden einer Datenquelle	1068
26.8.2	Eine Auswahlabfrage im DataGridView anzeigen	1072
26.8.3	Master-Detailbeziehungen im DataGrid anzeigen	1075
26.8.4	Datenbindung Chart-Control	1076

27	Erweiterte Grafikausgabe	1081
27.1	Transformieren mit der Matrix-Klasse	1081
27.1.1	Übersicht	1081
27.1.2	Translation	1082
27.1.3	Skalierung	1082
27.1.4	Rotation	1083
27.1.5	Scherung	1083
27.1.6	Zuweisen der Matrix	1084
27.2	Low-Level-Grafikmanipulationen	1084
27.2.1	Worauf zeigt Scan0?	1085
27.2.2	Anzahl der Spalten bestimmen	1086
27.2.3	Anzahl der Zeilen bestimmen	1087
27.2.4	Zugriff im Detail (erster Versuch)	1087
27.2.5	Zugriff im Detail (zweiter Versuch)	1089
27.2.6	Invertieren	1091
27.2.7	In Graustufen umwandeln	1092
27.2.8	Heller/Dunkler	1093
27.2.9	Kontrast	1094
27.2.10	Gamma-Wert	1095
27.2.11	Histogramm spreizen	1096
27.2.12	Ein universeller Grafikfilter	1098
27.3	Fortgeschrittene Techniken	1102
27.3.1	Flackerfrei dank Double Buffering	1102
27.3.2	Animationen	1104
27.3.3	Animated GIFs	1107
27.3.4	Auf einzelne GIF-Frames zugreifen	1110
27.3.5	Transparenz realisieren	1111
27.3.6	Eine Grafik maskieren	1113
27.3.7	JPEG-Qualität beim Sichern bestimmen	1114
27.4	Grundlagen der 3D-Vektorgrafik	1115
27.4.1	Datentypen für die Verwaltung	1116
27.4.2	Eine universelle 3D-Grafik-Klasse	1117
27.4.3	Grundlegende Betrachtungen	1118
27.4.4	Translation	1121
27.4.5	Streckung/Skalierung	1121
27.4.6	Rotation	1122
27.4.7	Die eigentlichen Zeichenroutinen	1124

27.5	Und doch wieder GDI-Funktionen ...	1127
27.5.1	Am Anfang war das Handle	1127
27.5.2	Gerätekontext (Device Context Types)	1129
27.5.3	Koordinatensysteme und Abbildungsmodi	1131
27.5.4	Zeichenwerkzeuge/Objekte	1136
27.5.5	Bitmaps	1138
27.6	Praxisbeispiele	1142
27.6.1	Die Transformationsmatrix verstehen	1142
27.6.2	Eine 3D-Grafikausgabe in Aktion	1145
27.6.3	Einen Fenster-Screenshot erzeugen	1148
28	Ressourcen/Lokalisierung	1151
28.1	Manifestressourcen	1151
28.1.1	Erstellen von Manifestressourcen	1151
28.1.2	Zugriff auf Manifestressourcen	1152
28.2	Typisierte Ressourcen	1154
28.2.1	Erzeugen von .resources-Dateien	1154
28.2.2	Hinzufügen der .resources-Datei zum Projekt	1155
28.2.3	Zugriff auf die Inhalte von .resources-Dateien	1155
28.2.4	ResourceManager direkt aus der .resources-Datei erzeugen	1156
28.2.5	Was sind .resx-Dateien?	1157
28.3	Streng typisierte Ressourcen	1157
28.3.1	Erzeugen streng typisierter Ressourcen	1157
28.3.2	Verwenden streng typisierter Ressourcen	1158
28.3.3	Streng typisierte Ressourcen per Reflection auslesen	1158
28.4	Anwendungen lokalisieren	1161
28.4.1	Localizable und Language	1161
28.4.2	Beispiel "Landesfahnen"	1161
28.4.3	Einstellen der aktuellen Kultur zur Laufzeit	1164
28.5	Praxisbeispiel	1166
28.5.1	Betrachter für Manifestressourcen	1166
29	Komponentenentwicklung	1171
29.1	Überblick	1171
29.2	Benutzersteuerelement	1172
29.2.1	Entwickeln einer Auswahl-ListBox	1172
29.2.2	Komponente verwenden	1174

29.3	Benutzerdefiniertes Steuerelement	1175
29.3.1	Entwickeln eines BlinkLabels	1175
29.3.2	Verwenden der Komponente	1177
29.4	Komponentenklasse	1178
29.5	Eigenschaften	1179
29.5.1	Einfache Eigenschaften	1179
29.5.2	Schreib-/Lesezugriff (Get/Set)	1179
29.5.3	Nur Lese-Eigenschaft (ReadOnly)	1180
29.5.4	Nur-Schreibzugriff (WriteOnly)	1180
29.5.5	Hinzufügen von Beschreibungen	1181
29.5.6	Ausblenden im Eigenschaftenfenster	1181
29.5.7	Einfügen in Kategorien	1182
29.5.8	Default-Wert einstellen	1182
29.5.9	Standard-Eigenschaft	1183
29.5.10	Wertebereichsbeschränkung und Fehlerprüfung	1184
29.5.11	Eigenschaften von Aufzählungstypen	1185
29.5.12	Standard Objekt-Eigenschaften	1186
29.5.13	Eigene Objekt-Eigenschaften	1187
29.6	Methoden	1189
29.6.1	Konstruktor	1190
29.6.2	Class-Konstruktor	1191
29.6.3	Destruktor	1192
29.6.4	Aufruf des Basisklassen-Konstruktors	1193
29.6.5	Aufruf von Basisklassen-Methoden	1193
29.7	Ereignisse (Events)	1193
29.7.1	Ereignis mit Standardargument definieren	1194
29.7.2	Ereignis mit eigenen Argumenten	1195
29.7.3	Ein Default-Ereignis festlegen	1196
29.7.4	Mit Ereignissen auf Windows-Messages reagieren	1196
29.8	Weitere Themen	1198
29.8.1	Wohin mit der Komponente?	1198
29.8.2	Assembly-Informationen festlegen	1199
29.8.3	Assemblies signieren	1201
29.8.4	Komponenten-Ressourcen einbetten	1202
29.8.5	Der Komponente ein Icon zuordnen	1202
29.8.6	Den Designmodus erkennen	1203
29.8.7	Komponenten lizenzieren	1203

29.9	Praxisbeispiele	1208
29.9.1	AnimGif – Komponente für die Anzeige von Animationen	1208
29.9.2	Eine FontComboBox entwickeln	1210
29.9.3	Das PropertyGrid verwenden	1212

Teil IV: WPF-Anwendungen

30	Einführung in WPF	1217
30.1	Einführung	1218
30.1.1	Was kann eine WPF-Anwendung?	1218
30.1.2	Die eXtensible Application Markup Language	1220
30.1.3	Verbinden von XAML und Basic-Code	1224
30.1.4	Zielplattformen	1231
30.1.5	Applikationstypen	1231
30.1.6	Vorteile und Nachteile von WPF-Anwendungen	1232
30.1.7	Weitere Dateien im Überblick	1233
30.2	Alles beginnt mit dem Layout	1235
30.2.1	Allgemeines zum Layout	1235
30.2.2	Positionieren von Steuerelementen	1238
30.2.3	Canvas	1241
30.2.4	StackPanel	1242
30.2.5	DockPanel	1244
30.2.6	WrapPanel	1246
30.2.7	UniformGrid	1246
30.2.8	Grid	1247
30.2.9	ViewBox	1252
30.2.10	TextBlock	1253
30.3	Das WPF-Programm	1256
30.3.1	Die Application-Klasse	1257
30.3.2	Das Startobjekt festlegen	1257
30.3.3	Kommandozeilenparameter verarbeiten	1259
30.3.4	Die Anwendung beenden	1259
30.3.5	Auswerten von Anwendungsereignissen	1260
30.4	Die Window-Klasse	1261
30.4.1	Position und Größe festlegen	1261
30.4.2	Rahmen und Beschriftung	1261
30.4.3	Das Fenster-Icon ändern	1262

30.4.4	Anzeige weiterer Fenster	1262
30.4.5	Transparenz	1262
30.4.6	Abstand zum Inhalt festlegen	1263
30.4.7	Fenster ohne Fokus anzeigen	1264
30.4.8	Ereignisfolge bei Fenstern	1264
30.4.9	Ein paar Worte zur Schriftdarstellung	1265
30.4.10	Ein paar Worte zur Controldarstellung	1267
30.4.11	Wird mein Fenster komplett mit WPF gerendert?	1269
31	Übersicht WPF-Controls	1271
31.1	Allgemeingültige Eigenschaften	1271
31.2	Label	1273
31.3	Button, RepeatButton, ToggleButton	1273
31.3.1	Schaltflächen für modale Dialoge	1274
31.3.2	Schaltflächen mit Grafik	1275
31.4	TextBox, PasswortBox	1276
31.4.1	TextBox	1276
31.4.2	PasswordBox	1278
31.5	CheckBox	1279
31.6	RadioButton	1281
31.7	ListBox, ComboBox	1282
31.7.1	ListBox	1282
31.7.2	ComboBox	1285
31.7.3	Den Content formatieren	1286
31.8	Image	1288
31.8.1	Grafik per XAML zuweisen	1288
31.8.2	Grafik zur Laufzeit zuweisen	1288
31.8.3	Bild aus Datei laden	1289
31.8.4	Die Grafiskalierung beeinflussen	1290
31.9	MediaElement	1291
31.10	Slider, ScrollBar	1294
31.10.1	Slider	1294
31.10.2	ScrollBar	1295
31.11	ScrollViewer	1296
31.12	Menu, ContextMenu	1297
31.12.1	Menu	1297
31.12.2	Tastenkürzel	1298
31.12.3	Grafiken	1299

31.12.4	Weitere Möglichkeiten	1300
31.12.5	ContextMenu	1301
31.13	ToolBar	1302
31.14	StatusBar, ProgressBar	1305
31.14.1	StatusBar	1305
31.14.2	ProgressBar	1307
31.15	Border, GroupBox, BulletDecorator	1308
31.15.1	Border	1308
31.15.2	GroupBox	1309
31.15.3	BulletDecorator	1310
31.16	RichTextBox	1312
31.16.1	Verwendung und Anzeige von vordefiniertem Text	1312
31.16.2	Neues Dokument zur Laufzeit erzeugen	1314
31.16.3	Sichern von Dokumenten	1314
31.16.4	Laden von Dokumenten	1316
31.16.5	Texte per Code einfügen/modifizieren	1317
31.16.6	Texte formatieren	1318
31.16.7	EditingCommands	1320
31.16.8	Grafiken/Objekte einfügen	1320
31.16.9	Rechtschreibkontrolle	1322
31.17	FlowDocumentPageViewer, -Reader, -ScrollViewer	1322
31.17.1	FlowDocumentPageViewer	1322
31.17.2	FlowDocumentReader	1323
31.17.3	FlowDocumentScrollViewer	1323
31.18	FlowDocument	1323
31.18.1	FlowDocument per XAML beschreiben	1324
31.18.2	FlowDocument per Code erstellen	1326
31.19	DocumentViewer	1327
31.20	Expander, TabControl	1328
31.20.1	Expander	1328
31.20.2	TabControl	1330
31.21	Popup	1331
31.22	TreeView	1333
31.23	ListView	1336
31.24	DataGrid	1337
31.25	Calendar/DatePicker	1338
31.26	InkCanvas	1342
31.26.1	Stift-Parameter definieren	1342

31.26.2	Die Zeichenmodi	1343
31.26.3	Inhalte laden und sichern	1344
31.26.4	Konvertieren in eine Bitmap	1344
31.26.5	Weitere Eigenschaften	1345
31.27	Ellipse, Rectangle, Line und Co.	1345
31.27.1	Ellipse	1346
31.27.2	Rectangle	1346
31.27.3	Line	1347
31.28	Browser	1347
31.29	Ribbon	1349
31.30	Chart	1351
31.31	WindowsFormsHost	1352

32 Wichtige WPF-Techniken 1355

32.1	Eigenschaften	1355
32.1.1	Abhängige Eigenschaften (Dependency Properties)	1355
32.1.2	Angehängte Eigenschaften (Attached Properties)	1356
32.2	Einsatz von Ressourcen	1357
32.2.1	Was sind eigentlich Ressourcen?	1357
32.2.2	Wo können Ressourcen gespeichert werden?	1357
32.2.3	Wie definiere ich eine Ressource?	1359
32.2.4	Statische und dynamische Ressourcen	1360
32.2.5	Wie werden Ressourcen adressiert?	1361
32.2.6	System-Ressourcen einbinden	1362
32.3	Das WPF-Ereignis-Modell	1362
32.3.1	Einführung	1362
32.3.2	Routed Events	1363
32.3.3	Direkte Events	1365
32.4	Verwendung von Commands	1365
32.4.1	Einführung Commands	1366
32.4.2	Verwendung vordefinierter Commands	1366
32.4.3	Das Ziel des Commands	1368
32.4.4	Welche vordefinierten Commands stehen zur Verfügung?	1369
32.4.5	Commands an Ereignismethoden binden	1369
32.4.6	Wie kann ich ein Command per Code auslösen?	1371
32.4.7	Command-Ausführung verhindern	1371
32.5	Das WPF-Style-System	1372
32.5.1	Übersicht	1372

32.5.2	Benannte Styles	1372
32.5.3	Typ-Styles	1374
32.5.4	Styles anpassen und vererben	1375
32.6	Verwenden von Triggern	1377
32.6.1	Eigenschaften-Trigger (Property triggers)	1378
32.6.2	Ereignis-Trigger	1380
32.6.3	Daten-Trigger	1381
32.7	Einsatz von Templates	1381
32.8	Transformationen, Animationen und StoryBoards	1387
32.8.1	Transformationen	1387
32.8.2	Animationen mit dem StoryBoard realisieren	1392
33	Grundlagen der WPF-Datenbindung	1397
33.1	Grundprinzip	1397
33.1.1	Bindungsarten	1398
33.1.2	Wann wird eigentlich die Quelle aktualisiert?	1399
33.1.3	Bindung zur Laufzeit realisieren	1400
33.2	Binden an Objekte	1402
33.2.1	Objekte im Code instanziiieren	1402
33.2.2	Verwenden der Instanz im VB-Quellcode	1404
33.2.3	Anforderungen an die Quell-Klasse	1404
33.2.4	Instanziiieren von Objekten per VB-Code	1406
33.3	Binden von Collections	1407
33.3.1	Anforderung an die Collection	1407
33.3.2	Einfache Anzeige	1408
33.3.3	Navigation zwischen den Objekten	1409
33.3.4	Einfache Anzeige in einer ListBox	1411
33.3.5	DataTemplates zur Anzeigeformatierung	1412
33.3.6	Mehr zu List- und ComboBox	1413
33.3.7	Verwendung der ListView	1415
33.4	Anzeige von Datenbankinhalten	1417
33.4.1	Datenmodell per LINQ to SQL-Designer erzeugen	1417
33.4.2	Die Programm-Oberfläche	1418
33.4.3	Der Zugriff auf die Daten	1419
33.5	Noch einmal zurück zu den Details	1420
33.5.1	Navigieren in den Daten	1421
33.5.2	Sortieren	1422
33.5.3	Filtern	1423

33.6	Drag & Drop-Datenbindung	1423
33.6.1	Vorgehensweise	1423
33.6.2	Weitere Möglichkeiten	1426
33.7	Formatieren von Werten	1427
33.7.1	IValueConverter	1428
33.7.2	BindingBase.StringFormat-Eigenschaft	1430
33.8	Das DataGrid als Universalwerkzeug	1431
33.8.1	Grundlagen der Anzeige	1431
33.8.2	Vom Betrachten zum Editieren	1436
34	Drucken/Druckvorschau mit WPF	1437
34.1	Grundlagen	1437
34.1.1	XPS-Dokumente	1437
34.1.2	System.Printing	1438
34.1.3	System.Windows.Xps	1439
34.2	Einfache Druckausgaben mit dem PrintDialog	1439
34.3	Mehrseitige Dokumente und Druckvorschau-Funktion	1442
34.3.1	Fix-Dokumente	1442
34.3.2	Flow-Dokumente	1448
34.4	Druckerinfos und Druckerauswahl/-konfiguration	1451
34.4.1	Die installierten Drucker bestimmen	1452
34.4.2	Den Standarddrucker bestimmen	1453
34.4.3	Mehr über einzelne Drucker erfahren	1453
34.4.4	Spezifische Druckeinstellungen vornehmen	1455
34.4.5	Direkte Druckausgabe	1457
35	WPF-Entwicklung	1459
35.1	Entwicklungstools für WPF-Anwendungen	1459
35.1.1	Microsoft Visual Studio 2010	1459
35.1.2	Microsoft Expression Blend	1460
35.1.3	Weitere Tools	1464
35.2	Distribution	1464
35.2.1	.NET Framework 4 Client Profil	1465
35.2.2	Was fehlt im Client-Profil?	1466
35.2.3	Installationsgröße weiter verringern	1466

36 Silverlight-Entwicklung	1467
36.1 Einführung	1467
36.1.1 Zielplattformen	1468
36.1.2 Silverlight-Applikationstypen	1468
36.1.3 Wichtige Unterschiede zu den WPF-Anwendungen	1470
36.1.4 Vor- und Nachteile von Silverlight-Anwendungen	1472
36.1.5 Entwicklungstools	1474
36.1.6 Installation auf dem Client	1474
36.2 Die Silverlight-Anwendung im Detail	1476
36.2.1 Ein kleines Beispielprojekt	1476
36.2.2 Das Application Package und das Test-Web	1478
36.3 Die Projektdateien im Überblick	1482
36.3.1 Projektverwaltung mit App.xaml & App.xaml.vb	1482
36.3.2 MainPage.xaml & MainPage.xaml.vb	1484
36.3.3 AssemblyInfo.vb	1485
36.4 Fenster und Seiten in Silverlight	1485
36.4.1 Das Standardfenster	1486
36.4.2 Untergeordnete Silverlight-Fenster	1487
36.4.3 UserControls für die Anzeige von Detaildaten verwenden	1489
36.4.4 Navigieren in Silverlight-Anwendungen	1490
36.5 Datenbanken/Datenbindung	1495
36.5.1 ASP.NET Webdienste/WCF Dienste	1496
36.5.2 WCF Data Services	1504
36.6 Isolierter Speicher	1515
36.6.1 Grundkonzept	1515
36.6.2 Das virtuelle Dateisystem verwalten	1516
36.6.3 Arbeiten mit Dateien	1519
36.7 Praxisbeispiele	1520
36.7.1 Eine Out-of-Browser-Applikation realisieren	1520
36.7.2 Out-of-Browser-Anwendung aktualisieren	1524
36.7.3 Testen, ob die Anwendung mit dem Internet verbunden ist	1525
36.7.4 Auf Out-of-Browser-Anwendung testen	1526
36.7.5 Den Browser bestimmen	1526
36.7.6 Parameter an das Plug-in übergeben	1527
36.7.7 Auf den QueryString zugreifen	1529
36.7.8 Timer in Silverlight nutzen	1530
36.7.9 Dateien lokal speichern	1531

Teil IV: ASP.NET-Anwendungen

37 ASP.NET-Einführung	1535
37.1 ASP.NET für Ein- und Umsteiger	1535
37.1.1 ASP – der Blick zurück	1535
37.1.2 Was ist bei ASP.NET anders?	1536
37.1.3 Was gibt es noch in ASP.NET?	1538
37.1.4 Vorteile von ASP.NET gegenüber ASP	1539
37.1.5 Voraussetzungen für den Einsatz von ASP.NET	1540
37.1.6 Und was hat das alles mit Visual Basic zu tun?	1541
37.2 Unsere erste Web-Anwendung	1543
37.2.1 Visueller Entwurf der Bedienoberfläche	1543
37.2.2 Zuweisen der Objekteigenschaften	1546
37.2.3 Verknüpfen der Objekte mit Ereignissen	1547
37.2.4 Programm kompilieren und testen	1548
37.3 Die ASP.NET-Projektdateien	1549
37.3.1 Die ASP.NET-Projekttypen	1549
37.3.2 ASPX-Datei(en)	1551
37.3.3 Die aspx.vb-Datei(en)	1554
37.3.4 Die Datei Global.asax	1554
37.3.5 Das Startformular	1555
37.3.6 Die Datei Web.config	1555
37.3.7 Masterpages (master-Dateien)	1558
37.3.8 Sitemap (Web.sitemap)	1558
37.3.9 Benutzersteuerelemente (ascx-Dateien)	1559
37.3.10 Die Web-Projekt-Verzeichnisse	1559
37.4 Lernen am Beispiel	1560
37.4.1 Erstellen des Projekts	1560
37.4.2 Oberflächengestaltung	1561
37.4.3 Ereignisprogrammierung	1562
37.4.4 Ein Fehler, was nun?	1563
37.4.5 Ereignisse von Textboxen	1565
37.4.6 Ein gemeinsamer Ereignis-Handler	1565
37.4.7 Eingabefokus setzen	1566
37.4.8 Ausgaben in einer Tabelle	1566
37.4.9 Scrollen der Anzeige	1569
37.4.10 Zusammenspiel mehrerer Formulare	1569
37.4.11 Umleiten bei Direktaufruf	1571

37.4.12	Ärger mit den Cookies	1572
37.4.13	Export auf den IIS	1573

38 Übersicht ASP.NET-Controls 1575

38.1	Einfache Steuerelemente im Überblick	1575
38.1.1	Label	1575
38.1.2	TextBox	1577
38.1.3	Button, ImageButton, LinkButton	1578
38.1.4	CheckBox, RadioButton	1579
38.1.5	CheckBoxList, BulletList, RadioButtonList	1580
38.1.6	Table	1581
38.1.7	Hyperlink	1583
38.1.8	Image, ImageMap	1583
38.1.9	Calendar	1585
38.1.10	Panel	1586
38.1.11	HiddenField	1586
38.1.12	Substitution	1587
38.1.13	XML	1588
38.1.14	FileUpload	1590
38.1.15	AdRotator	1591
38.2	Steuerelemente für die Seitennavigation	1592
38.2.1	Mehr Übersicht mit Web.Sitemap	1592
38.2.2	Menu	1594
38.2.3	TreeView	1597
38.2.4	SiteMapPath	1600
38.2.5	MultiView, View	1601
38.2.6	Wizard	1602
38.3	Webseitenlayout/-design	1604
38.3.1	Masterpages	1604
38.3.2	Themes/Skins	1607
38.3.3	Webparts	1610
38.4	Die Validator-Controls	1611
38.4.1	Übersicht	1611
38.4.2	Wo findet die Fehlerprüfung statt?	1612
38.4.3	Verwendung	1612
38.4.4	RequiredFieldValidator	1613
38.4.5	CompareValidator	1614
38.4.6	RangeValidator	1616

38.4.7	RegularExpressionValidator	1616
38.4.8	CustomValidator	1617
38.4.9	ValidationSummary	1620
38.4.10	Weitere Möglichkeiten	1621
38.5	Praxisbeispiele	1621
38.5.1	Themes und Skins verstehen	1621
38.5.2	Masterpages verwenden	1626
38.5.3	Webparts verwenden	1629
39	ASP.NET-Datenbindung	1635
39.1	Alt und Neu im Vergleich	1635
39.1.1	Variante 1: Konventionelle Programmierung	1635
39.1.2	Variante 2: Mit DataSource	1638
39.2	Einführung	1641
39.2.1	Konzept	1642
39.2.2	Übersicht DataSource-Steuerelemente	1643
39.3	SQLDataSource	1644
39.3.1	Datenauswahl mit Parametern	1646
39.3.2	Parameter für INSERT, UPDATE und DELETE	1647
39.3.3	Methoden	1648
39.3.4	Caching	1650
39.3.5	Aktualisieren/Refresh	1650
39.4	AccessDataSource	1650
39.5	ObjectDataSource	1650
39.5.1	Verbindung zwischen Objekt und DataSource	1651
39.5.2	Ein Beispiel sorgt für Klarheit	1652
39.5.3	Geschäftsobjekte in einer Session verwalten	1656
39.6	SitemapDataSource	1658
39.7	LinqDataSource	1659
39.7.1	Bindung von einfachen Collections	1660
39.7.2	Bindung eines LINQ to SQL-DataContext	1661
39.8	EntityDataSource	1663
39.8.1	Entity Data Model erstellen	1663
39.8.2	EntityDataSource anbinden	1666
39.8.3	Datenmenge filtern	1668
39.9	XmlDataSource	1669

39.10	QueryExtender	1670
39.10.1	Grundlagen	1671
39.10.2	Suchen	1672
39.10.3	Sortieren	1674
39.11	GridView	1675
39.11.1	Auswahlfunktion (Zeilenauswahl)	1676
39.11.2	Auswahl mit mehrspaltigem Index	1676
39.11.3	Hyperlink-Spalte für Detailansicht	1677
39.11.4	Spalten erzeugen	1678
39.11.5	Paging realisieren	1679
39.11.6	Edit, Update, Delete	1680
39.11.7	Keine Daten, was tun?	1681
39.12	DetailsView	1681
39.13	FormView	1683
39.14	DataList	1686
39.14.1	Bearbeitungsfunktionen implementieren	1686
39.14.2	Layout verändern	1688
39.15	Repeater	1688
39.16	ListView	1690
39.17	Chart	1690

40 ASP.NET-Objekte und Techniken 1693

40.1	Wichtige ASP.NET-Objekte	1693
40.1.1	HTTPApplication	1693
40.1.2	Application	1696
40.1.3	Session	1697
40.1.4	Page	1699
40.1.5	Request	1702
40.1.6	Response	1705
40.1.7	Server	1709
40.1.8	Cookies verwenden	1710
40.2	ASP.NET-Fehlerbehandlung	1713
40.2.1	Fehler beim Entwurf	1713
40.2.2	Laufzeitfehler	1713
40.2.3	Eine eigene Fehlerseite	1715
40.2.4	Fehlerbehandlung im Web Form	1716
40.2.5	Fehlerbehandlung in der Anwendung	1717

40.2.6	Alternative Fehlerseite einblenden	1718
40.2.7	Lokale Fehlerbehandlung	1719
40.2.8	Seite nicht gefunden! – Was nun?	1719
40.3	E-Mail-Versand in ASP.NET	1720
40.3.1	Übersicht	1720
40.3.2	Mail-Server bestimmen	1721
40.3.3	Einfache Text-E-Mails versenden	1722
40.3.4	E-Mails mit Dateianhang	1723
40.4	Sicherheit von Webanwendungen	1724
40.4.1	Authentication	1724
40.4.2	Forms Authentication realisieren	1726
40.4.3	Impersonation	1729
40.4.4	Authorization	1730
40.4.5	Administrieren der Website	1732
40.4.6	Steuerelemente für das Login-Handling	1736
40.4.7	Programmieren der Sicherheitseinstellungen	1741
40.5	Die Verwendung von AJAX in ASP.NET-Anwendungen	1743
40.5.1	Was ist eigentlich AJAX und was kann es?	1743
40.5.2	Die AJAX-Controls	1744
40.5.3	AJAX-Control-Toolkit	1747
40.6	User Controls/Webbenutzersteuerelemente	1748
40.6.1	Ein simples Einstiegsbeispiel	1749
40.6.2	Dynamische Grafiken in einem User Control anzeigen	1753
40.6.3	Grafikausgaben per User Control realisieren	1757

Anhang

A	Glossar	1763
B	Wichtige Datei-Extensions	1769
	Index	1771