

Inhalt

Vorwort zur 5. Auflage

27

1 Allgemeine Einführung in .NET 29

1.1	Warum .NET?	29
1.1.1	Ein paar Worte zu diesem Buch	32
1.1.2	Die Beispielprogramme	34
1.2	.NET unter die Lupe genommen	34
1.2.1	Das Entwicklerdilemma	34
1.2.2	.NET – ein paar allgemeine Eigenschaften	36
1.2.3	Das Sprachenkonzept	37
1.2.4	Die Common Language Specification (CLS)	38
1.2.5	Das Common Type System (CTS)	40
1.2.6	Das .NET Framework	41
1.2.7	Die Common Language Runtime (CLR)	42
1.2.8	Die .NET-Klassenbibliothek	43
1.2.9	Das Konzept der Namespaces	44
1.3	Assemblys	45
1.3.1	Die Metadaten	46
1.3.2	Das Manifest	47
1.4	Die Entwicklungsumgebung	47
1.4.1	Editionen von Visual Studio 2010	48
1.4.2	Hard- und Softwareanforderungen	48
1.4.3	Installation	49
1.4.4	Entwicklungsumgebung von Visual Studio 2010	49

2 Grundlagen der Sprache C# 55

2.1	Konsolenanwendungen	55
2.1.1	Allgemeine Anmerkungen	55
2.1.2	Ein erstes Konsolenprogramm	56
2.2	Grundlagen der C#-Syntax	59
2.2.1	Kennzeichnen, dass eine Anweisung abgeschlossen ist.....	59
2.2.2	Anweisungs- und Gliederungsblöcke	60
2.2.3	Kommentare	61
2.2.4	Groß- und Kleinschreibung	62
2.2.5	Struktur einer Konsolenanwendung	63
2.3	Variablen und Datentypen	65
2.3.1	Variablendeklaration	65

5

2.3.2	Variablenbezeichner	67
2.3.3	Zugriff auf eine Variable	68
2.3.4	Ein- und Ausgabemethoden der Klasse »Console«	68
2.3.5	Einfache Datentypen	74
2.3.6	Typkonvertierung	85
2.4	Operatoren	94
2.4.1	Arithmetische Operatoren	94
2.4.2	Vergleichsoperatoren	97
2.4.3	Logische Operatoren	98
2.4.4	Bitweise Operatoren	101
2.4.5	Zuweisungsoperatoren	104
2.4.6	Stringverkettung	105
2.4.7	Sonstige Operatoren	105
2.4.8	Operator-Vorrangregeln	106
2.5	Datenfelder (Arrays)	107
2.5.1	Deklaration und Initialisierung eines Arrays	107
2.5.2	Zugriff auf Array-Elemente	108
2.5.3	Speicherabbild eines Arrays	110
2.5.4	Mehrdimensionale Arrays	112
2.5.5	Array-Größe zur Laufzeit festlegen	114
2.5.6	Bestimmung der Array-Obergrenze	115
2.5.7	Gesamtanzahl der Array-Elemente	116
2.5.8	Verzweigte Arrays	116
2.6	Kontrollstrukturen	118
2.6.1	if-Anweisung	118
2.6.2	switch-Statement	124
2.7	Programmschleifen	129
2.7.1	for-Schleife	130
2.7.2	foreach-Schleife	141
2.7.3	do- und while-Schleife	142

3 Klassendesign 149

3.1	Einführung in die Objektorientierung	149
3.1.1	Vorteile der objektorientierten Programmierung	151
3.2	Klassendefinition	152
3.2.1	Klassen in Visual Studio anlegen	152
3.2.2	Das Projekt »GeometricObjectsSolution«	153
3.2.3	Deklaration von Objektvariablen	154
3.2.4	Zugriffsmodifizierer einer Klasse	155
3.2.5	Splitten einer Klassendefinition mit »partial«	156

3.3	Arbeiten mit Objektreferenzen	156
3.3.1	Prüfen auf Initialisierung	156
3.3.2	Ein Objekt freigeben	158
3.3.3	Mehrere Referenzen auf ein Objekt	158
3.4	Referenz- und Wertetypen	159
3.5	Eigenschaften eines Objekts	160
3.5.1	Öffentliche Felder	160
3.5.2	Datenkapselung mit Eigenschaftsmethoden sicherstellen	162
3.5.3	Ergänzung der Klasse »Circle«	164
3.5.4	Lese- und schreibgeschützte Eigenschaften	165
3.5.5	Sichtbarkeit der Accessoren »get« und »set«	166
3.5.6	Unterstützung von Visual Studio 2010	167
3.5.7	Automatisch implementierte Eigenschaften	167
3.5.8	Vereinfachte Objektinstanziierung mit Objektinitialisierern	168
3.6	Methoden eines Objekts	169
3.6.1	Methoden mit Rückgabewert	169
3.6.2	Methoden ohne Rückgabewert	172
3.6.3	Methoden mit Parameterliste	173
3.6.4	Methodenüberladung	175
3.6.5	Variablen innerhalb einer Methode (Lokale Variablen)	177
3.6.6	Referenz- und Wertparameter	178
3.6.7	Zugriff auf private Daten	190
3.6.8	Namenskonflikte mit »this« lösen	191
3.6.9	Trennung von Daten und Code im Speicher	192
3.6.10	Methode oder Eigenschaft?	193
3.6.11	Methoden und Eigenschaften umbenennen	194
3.7	Konstruktoren	195
3.7.1	Konstruktoren bereitstellen	196
3.7.2	Parametrisierte Konstruktoren und die Objektinitialisierung	197
3.7.3	Konstruktoraufrufe	197
3.7.4	Definition von Konstruktoren	198
3.7.5	»internal«-Konstruktoren	199
3.7.6	»private«-Konstruktoren	199
3.7.7	Konstruktorenaufrufe umleiten	199
3.8	Der Destruktor	200
3.9	Konstanten in einer Klasse	201
3.9.1	Konstanten mit dem Schlüsselwort »const«	201
3.9.2	Schreibgeschützte Felder mit »readonly«	202
3.10	Statische Klassenkomponenten	203
3.10.1	Begrifflichkeiten	203
3.10.2	Statische Klassenvariable in der Klasse »Circle«	204

3.10.3	Klassenspezifische Methoden	205
3.10.4	Statische Konstruktoren (Klasseninitialisierer)	206
3.10.5	Statische Klassen	207
3.10.6	Stand der Klasse »Circle«	208

4	Vererbung, Polymorphie und Interfaces	211
4.1	Die Vererbung	211
4.1.1	Basisklassen und abgeleitete Klassen	211
4.1.2	Ableitung einer Klasse	212
4.1.3	Klassen, die nicht abgeleitet werden können	214
4.1.4	Konstruktoren in abgeleiteten Klassen	214
4.1.5	Der Zugriffsmodifizierer »protected«	215
4.1.6	Konstruktorverkettung in der Vererbung	216
4.2	Der Problemfall geerbter Methoden	220
4.2.1	Geerbte Methoden mit »new« verdecken	222
4.2.2	Abstrakte Methoden	224
4.2.3	Virtuelle Methoden	226
4.3	Typumwandlung und Typuntersuchung von Objektvariablen	228
4.3.1	Implizite Typumwandlung von Objektreferenzen	228
4.3.2	Explizite Typumwandlung von Objektreferenzen	230
4.3.3	Typuntersuchung mit dem is-Operator	230
4.3.4	Typumwandlung mit dem as-Operator	231
4.4	Polymorphie	232
4.4.1	»Klassische« Methodenimplementierung	232
4.4.2	Abstrakte Methoden	233
4.4.3	Virtuelle Methoden	234
4.4.4	Versiegelte Methoden	238
4.4.5	Überladen einer Basisklassenmethode	238
4.4.6	Statische Member und Vererbung	239
4.5	Das Projekt »GeometricObjectsSolution« ergänzen	240
4.6	Hat-ein(e)-Beziehungen	243
4.6.1	Innere Klassen (Nested Classes)	249
4.7	Interfaces (Schnittstellen)	253
4.7.1	Schnittstellendefinition	254
4.7.2	Schnittstellenimplementierung	254
4.7.3	Interpretation der Schnittstellen	259
4.8	Das Zerstören von Objekten – der Garbage Collector	265
4.8.1	Arbeitsweise des Garbage Collectors	265
4.8.2	Expliziter Aufruf des Garbage Collectors	267
4.8.3	Der Destruktor	267

4.8.4	Die »IDisposable«-Schnittstelle	268
4.8.5	Ergänzungen in den Klassen »Circle« und »Rectangle«	270

5 Delegates und Ereignisse 273

5.1	Delegates	273
5.1.1	Einführung in das Prinzip der Delegates	273
5.1.2	Vereinfachter Aufruf eines Delegates	277
5.1.3	Multicast-Delegate	278
5.1.4	Anonyme Methoden	282
5.2	Ereignisse eines Objekts	283
5.2.1	Ereignis in einer Ereignisquelle ergänzen	284
5.2.2	Behandlung eines Ereignisses im Ereignisempfänger.....	286
5.2.3	Allgemeine Betrachtungen der Ereignishandlerregistrierung	288
5.2.4	Wenn der Ereignisempfänger ein Ereignis nicht behandelt	289
5.2.5	Ereignisse mit Übergabeparameter	290
5.2.6	Ereignisse in der Vererbung	294
5.2.7	Hinter die Kulissen geblickt	294
5.2.8	Änderungen im Projekt »GeometricObjects«	296

6 Weitere .NET-Datentypen 301

6.1	Strukturen	301
6.1.1	Definition einer Struktur	301
6.1.2	Initialisieren einer Struktur	302
6.1.3	Weitere Merkmale einer Struktur	303
6.1.4	Verschachtelte Strukturen	304
6.1.5	Änderung im Projekt »GeometricObjects«	305
6.2	Enumerationen (Aufzählungen)	307
6.2.1	Wertzuweisung an »enum«-Mitglieder	308
6.2.2	Alle Mitglieder einer Aufzählung durchlaufen	309

7 Weitere Möglichkeiten von C# 311

7.1	Namensräume (Namespaces)	311
7.1.1	Zugriff auf Namespaces	312
7.1.2	Die »using«-Direktive	314
7.1.3	Globaler Namespace	314
7.1.4	Vermeiden von Mehrdeutigkeiten	315
7.1.5	Namespaces festlegen	316
7.1.6	Der ::-Operator	317

7.2	Generics – generische Datentypen	319
7.2.1	Lösung mit einer generischen Klasse	321
7.2.2	Typparameter mit Constraints einschränken	323
7.2.3	Generische Methoden	326
7.2.4	Generics und Vererbung	327
7.2.5	Konvertierung von Generics	329
7.2.6	Generische Delegates	330
7.2.7	Generische Klassen in der .NET-Klassenbibliothek	330
7.3	Operatorüberladung	331
7.3.1	Syntax der Operatorüberladung	332
7.3.2	Operatorüberladungen in »GeometricObjectsSolution«	333
7.3.3	»true«- und »false«-Operatoren	337
7.3.4	Benutzerdefinierte Konvertierungen	339
7.4	Indexer	343
7.4.1	Überladen von Indexern	346
7.4.2	Parameterbehaftete Eigenschaften	348
7.5	Erweiterungsmethoden	351
7.6	Implizit typisierte Variablen (Typinferenz)	355
7.7	Lambda-Ausdrücke	356
7.8	Anonyme Typen	359
7.9	Nullable-Typen	359
7.10	Attribute	361
7.10.1	Das »Flags«-Attribut	361
7.10.2	Anmerkungen zu den Attributen	363
7.10.3	Benutzerdefinierte Attribute	364
7.10.4	Attribute auswerten	368
7.10.5	Festlegen der Assembly-Eigenschaften	371
7.11	Partielle Methoden	372
7.12	Dynamisches Binden	374
7.12.1	Eine kurze Analyse	375
7.12.2	Dynamische Objekte	376
7.13	Unsicherer (unsafe) Programmcode – Zeigertechnik in C#	377
7.13.1	Einführung	377
7.13.2	Das Schlüsselwort »unsafe«	378
7.13.3	Deklaration von Zeigern	378
7.13.4	Die »fixed«-Anweisung	379
7.13.5	Zeigerarithmetik	381
7.13.6	Der Operator »->«	381

8 Auflistungsklassen (Collections) 383

8.1	Die Listen des Namespace »System.Collections«	384
8.1.1	Die elementaren Schnittstellen der Auflistungsklassen	384
8.2	Die Klasse »ArrayList«	386
8.2.1	Einträge hinzufügen	387
8.2.2	Datenaustausch zwischen einem Array und einer ArrayList	390
8.2.3	Die Elemente einer ArrayList sortieren	390
8.2.4	Sortieren von Arrays mit »ArrayList.Adapter«	397
8.3	Die Klasse »Hashtable«	398
8.3.1	Methoden und Eigenschaften der Schnittstelle »IDictionary«	398
8.3.2	Beispielprogramm zur Klasse »Hashtable«	399
8.4	Die Klassen »Queue« und »Stack«	404
8.4.1	Die »Stack«-Klasse	404
8.5	Objektauflistungen im Überblick	407
8.6	Generische Auflistungsklassen	407
8.6.1	Die Interfaces der generischen Auflistungsklassen	408
8.6.2	Die generische Auflistungsklasse »List<T>«	408
8.7	Eigene Auflistungen mit »yield« durchlaufen	415

9 Fehlerbehandlung und Debugging 419

9.1	Die Behandlung von Laufzeitfehlern	419
9.1.1	Laufzeitfehler erkennen	420
9.1.2	Die Behandlung von Exceptions	422
9.1.3	Die »try...catch«-Anweisung	423
9.1.4	Behandlung mehrerer Exceptions	425
9.1.5	Die »finally«-Anweisung	427
9.1.6	Das Weiterleiten von Ausnahmen	428
9.1.7	Die Hierarchie der Exceptions	434
9.1.8	Die Reihenfolge der »catch«-Klauseln	435
9.1.9	Die Basisklasse »Exception«	437
9.1.10	Benutzerdefinierte Ausnahmen	437
9.2	Debuggen mit Programmcode	441
9.2.1	Einführung	441
9.2.2	Die Klasse »Debug«	441
9.2.3	Die Klasse »Trace«	445
9.2.4	Ablaufverfolgung mit »TraceListener«-Objekten	446
9.2.5	Steuerung der Protokollierung mit Schaltern	451
9.2.6	Bedingte Kompilierung	455

9.3	Debuggen mit Visual Studio 2010	457
9.3.1	Debuggen im Haltemodus	457
9.3.2	Das Direktfenster	460
9.3.3	Weitere Alternativen, um Variableninhalte zu prüfen	462

10 LINQ to Objects **465**

10.1	Einführung in LINQ	465
10.1.1	Grundlagen der LINQ-Erweiterungsmethoden	466
10.1.2	Verzögerte Ausführung	468
10.2	LINQ to Objects	469
10.2.1	Musterdaten	469
10.2.2	Allgemeine Syntax	471
10.2.3	Übersicht über die Abfrageoperatoren	472
10.2.4	Die »from«-Klausel	473
10.2.5	Der Restriktionsoperator »where«	475
10.2.6	Projektionsoperatoren	477
10.2.7	Sortieroperatoren	478
10.2.8	Gruppieren mit »GroupBy«	479
10.2.9	Verknüpfungen mit »Join«	481
10.2.10	Die Set-Operatoren-Familie	484
10.2.11	Die Familie der Aggregatoperatoren	485
10.2.12	Generierungsoperatoren	489
10.2.13	Quantifizierungsoperatoren	490
10.2.14	Aufteilungsoperatoren	491
10.2.15	Elementoperatoren	493

11 Multithreading und die Task Parallel Library (TPL) **497**

11.1	Überblick	497
11.2	Multithreading mit der Klasse »Thread«	498
11.2.1	Einführung in das Multithreading	498
11.2.2	Threadzustände und Prioritäten	498
11.2.3	Zusammenspiel mehrerer Threads	500
11.2.4	Die Entwicklung einer einfachen Multithreading-Anwendung....	500
11.2.5	Die Klasse »Thread«	503
11.2.6	Threadpools nutzen	511
11.2.7	Synchronisation von Threads	513
11.2.8	Der Monitor zur Synchronisation	515
11.2.9	Das Synchronisationsobjekt »Mutex«	522
11.2.10	Das Attribut »MethodImpl«	524
11.2.11	Grundlagen asynchroner Methodenaufrufe	524
11.2.12	Asynchroner Methodenaufwurf	525

11.2.13	Asynchroner Aufruf mit Rückgabewerten	529
11.2.14	Eine Klasse mit asynchronen Methodenaufrufen	532
11.3	Die TPL (Task Parallel Library)	536
11.3.1	Allgemeines zur Parallelisierung mit der TPL	536
11.3.2	Die Klasse »Parallel«	537
11.3.3	Die Klasse »Task«	542

12 Arbeiten mit Dateien und Streams **549**

12.1	Namespaces der Ein- bzw. Ausgabe	550
12.1.1	Das Behandeln von Ausnahmen bei E/A-Operationen	551
12.2	Laufwerke, Verzeichnisse und Dateien	551
12.2.1	Die Klasse »File«	552
12.2.2	Die Klasse »FileInfo«	557
12.2.3	Die Klassen »Directory« und »DirectoryInfo«	559
12.2.4	Die Klasse »Path«	563
12.2.5	Die Klasse »DriveInfo«	565
12.2.6	Die Klasse »SpecialDirectories«	566
12.3	Die »Stream«-Klassen	567
12.3.1	Die abstrakte Klasse »Stream«	567
12.3.2	Die von »Stream« abgeleiteten Klassen im Überblick	570
12.3.3	Die Klasse »FileStream«	571
12.4	Die Klassen »TextReader« und »TextWriter«	577
12.4.1	Die Klasse »StreamWriter«	578
12.4.2	Die Klasse »StreamReader«	582
12.5	Die Klassen »BinaryReader« und »BinaryWriter«	584
12.5.1	Komplexe binäre Dateien	586

13 Binäre Serialisierung **593**

13.1	Serialisierungsverfahren	594
13.2	Serialisierung mit »BinaryFormatter«	595

14 Einige wichtige .NET-Klassen **603**

14.1	Die Klasse »Object«	603
14.1.1	Referenzvergleiche mit »Equals« und »ReferenceEquals«	604
14.1.2	»ToString« und »GetType«	604
14.1.3	Die Methode »MemberwiseClone«	605
14.2	Die Klasse »String«	608
14.2.1	Erzeugen eines Strings	608
14.2.2	Eigenschaften von »String«	609

14.2.3	Methoden der Klasse »String«	609
14.2.4	Zusammenfassung der Klasse »String«	619
14.3	Die Klasse »StringBuilder«	621
14.3.1	Allgemeines	621
14.3.2	Kapazität eines »StringBuilder«-Objekts	621
14.3.3	Konstruktoren der Klasse »StringBuilder«	622
14.3.4	Eigenschaften der Klasse »StringBuilder«	622
14.3.5	Die Methoden der Klasse »StringBuilder«	623
14.3.6	Allgemeine Anmerkungen	625
14.4	Der Typ »DateTime«	626
14.4.1	Zeitspanne »Tick«	626
14.4.2	Konstruktoren von »DateTime«	627
14.4.3	Eigenschaften von »DateTime«	628
14.4.4	Methoden der Klasse »DateTime«	629
14.5	Die Klasse »TimeSpan«	631
14.6	Ausgabeformatierung	633
14.6.1	Formatierung mit der Methode »String.Format«.....	634
14.6.2	Formatierung mit der Methode »ToString«	637
14.6.3	Benutzerdefinierte Formatierung	638

15 Projektmanagement und Visual Studio 2010 **641**

15.1	Der Projekttyp »Klassenbibliothek«	641
15.1.1	Mehrere Projekte in einer Projektmappe verwalten	642
15.1.2	Die Zugriffsmodifizierer »public« und »internal«	643
15.1.3	Friend-Assemblys	643
15.1.4	Einbinden einer Klassenbibliothek	644
15.2	Assemblys	646
15.2.1	Konzept der Assemblys	646
15.2.2	Allgemeine Beschreibung privater und globaler Assemblys.....	648
15.2.3	Struktur einer Assembly	648
15.2.4	Globale Assemblys	653
15.3	Konfigurationsdateien	664
15.3.1	Einführung	664
15.3.2	Die verschiedenen Konfigurationsdateien	665
15.3.3	Struktur einer Anwendungskonfigurationsdatei.....	667
15.3.4	Anwendungskonfigurationsdatei mit Visual Studio 2010 bereitstellen	670
15.3.5	Einträge der Anwendungskonfigurationsdatei auswerten	671
15.3.6	Editierbare, anwendungsbezogene Einträge mit <appSettings>	676
15.4	Versionsumleitung in einer Konfigurationsdatei	678
15.4.1	Herausgeberrichtliniendatei	683

15.5	XML-Dokumentation	684
15.5.1	Prinzip der XML-Dokumentation	685
15.5.2	XML-Kommentar-Tags	687
15.5.3	Generieren der XML-Dokumentationsdatei	688
15.6	Der Klassendesigner (Class Designer)	690
15.6.1	Ein typisches Klassendiagramm	690
15.6.2	Hinzufügen und Ansicht von Klassendiagrammen	692
15.6.3	Die Toolbox des Klassendesigners	693
15.6.4	Das Fenster »Klassendetails«	693
15.6.5	Klassendiagramme als Bilder exportieren	695
15.7	Refactoring	695
15.7.1	Methode extrahieren	696
15.7.2	Bezeichner umbenennen	698
15.7.3	Felder enkapseln	699
15.8	Code Snippets (Codeausschnitte)	699
15.8.1	Codeausschnitte einfügen	700
15.8.2	Anatomie eines Codeausschnitts	700

16 XML

703

16.1	XML-Dokumente	703
16.1.1	Wohlgeformte und gültige XML-Dokumente	704
16.1.2	Regeln für wohlgeformten XML-Code	706
16.1.3	Kommentare	709
16.1.4	Verarbeitungsanweisungen	710
16.1.5	Reservierte Zeichen in XML	710
16.1.6	CDATA-Abschnitte	711
16.1.7	Namensräume (Namespaces)	711
16.2	Gültigkeit eines XML-Dokuments	718
16.2.1	XML Schema (XSD)	719
16.2.2	XML-Dokument mit einem XML Schema verknüpfen.....	720
16.2.3	Struktur eines XML Schemas	723
16.3	Die Klasse »XmlReader«	729
16.3.1	XML-Dokumente mit einem »XmlReader«-Objekt lesen.....	729
16.3.2	Validieren eines XML-Dokuments	736
16.4	Eigenschaften und Methoden der Klasse »XmlReader«	739
16.4.1	Navigation mit dem »XmlReader«	739
16.4.2	Eigenschaften und Methoden im Zusammenhang mit Attributen	740
16.4.3	Eigenschaften und Methoden im Zusammenhang mit Namespaces	740
16.4.4	Daten lesen	741

16.5	Die Klasse »XmlWriter«	741
16.5.1	Die Methoden der Klasse »XmlWriter«	746
16.6	Navigation durch XML (XPath)	746
16.6.1	Die Klasse »XPathNavigator«	746
16.6.2	XPath-Ausdrücke	751
16.6.3	Kontextknoten	752
16.6.4	Beispiele mit XPath-Ausdrücken	753
16.6.5	Knotenmengen mit der »Select«-Methode	756
16.6.6	Auswerten von XPath-Ausdrücken	760
16.7	Document Object Model (DOM)	764
16.7.1	Allgemeines	764
16.7.2	Arbeiten mit XmlDocument	765
16.7.3	XmlDocument und XPathNavigator	766
16.7.4	Die Klasse »XmlNode« (Operationen mit Knoten)	767
16.7.5	XML-Struktur manipulieren	775
16.7.6	Knoten ändern	776
16.7.7	Löschen in einem XML-Dokument	778
16.8	Serialisierung mit »XmlSerializer«	780
16.8.1	XML-Serialisierung mit Attributen steuern	782
16.9	LINQ to XML	785
16.9.1	Klassenhierarchie von LINQ to XML	785
16.9.2	Die Klasse »XElement«	786
16.9.3	Die Klasse »XDocument«	789
16.9.4	Navigation im XML-Dokument	790
16.9.5	Änderungen am XML-Dokument vornehmen.....	795

17 WPF – Die Grundlagen

797

17.1	Merkmale einer WPF-Anwendung	797
17.2	Anwendungstypen	799
17.3	Eine WPF-Anwendung und ihre Dateien	800
17.3.1	Die Datei »App.xaml«	802
17.3.2	Die Datei »App.xaml.cs«	802
17.3.3	Die Dateien »baml« und »g.cs«	803
17.4	Einführung in XAML	803
17.4.1	Struktur einer XAML-Datei	804
17.4.2	XAML-Elemente	805
17.4.3	Eigenschaften eines XAML-Elements festlegen.....	805
17.4.4	Typkonvertierung	809
17.4.5	Markup-Erweiterungen (Markup Extensions)	810
17.4.6	Namespaces	812

17.4.7	XAML-Spracherweiterungen	815
17.4.8	Markup-Erweiterungen	816
17.5	Abhängige und angehängte Eigenschaften	816
17.5.1	Abhängige Eigenschaften	816
17.5.2	Angehängte Eigenschaften	818
17.6	Logischer und visueller Elementbaum	818
17.6.1	Warum wird zwischen den Elementbäumen unterschieden?	820
17.6.2	Elementbäume mit Code ermitteln	820
17.7	Ereignisse in der WPF	822
17.7.1	Allgemeine Grundlagen	822
17.7.2	Routed Events	823

18 WPF-Containerelemente 829

18.1	WPF-Fenster	829
18.1.1	Nachrichtenfenster mit »MessageBox«	830
18.1.2	Die Methode »MessageBox.Show«	831
18.1.3	Fenster vom Typ »Window«	833
18.1.4	Fenster vom Typ »NavigationWindow«	837
18.1.5	Das »Page«-Element	838
18.1.6	Das »Frame«-Steuerelement	845
18.2	Layoutcontainer	846
18.2.1	Gemeinsame Eigenschaften der Layoutcontainer.....	847
18.3	Canvas	848
18.4	StackPanel	849
18.5	WrapPanel	852
18.6	DockPanel	853
18.7	UniformGrid	856
18.8	Grid	857
18.8.1	Struktur eines »Grid« festlegen	857
18.8.2	»ColumnSpan« und »RowSpan«	861
18.8.3	Spalten- und Zeilenbreite mit »GridSplitter« ändern	862
18.9	Verschachtelte Layoutcontainer	863

19 WPF-Steuerelemente 867

19.1	Hierarchie der WPF-Komponenten	867
19.2	Steuerelemente positionieren	869
19.2.1	Die Eigenschaften »Top«, »Bottom«, »Right« und »Left«	869
19.2.2	Außenrand mit der Eigenschaft »Margin« festlegen	869
19.2.3	Innenrand mit der Eigenschaft »Padding« festlegen	870

19.3	Allgemeine Eigenschaften der WPF-Komponenten	870
19.3.1	Die Eigenschaft »Content«	870
19.3.2	Größe einer Komponente	872
19.3.3	Ausrichtung einer Komponente	873
19.3.4	Die Eigenschaften »Padding« und »Margin«	874
19.3.5	Sichtbarkeit eines Steuerelements	876
19.3.6	Farbeinstellungen	877
19.3.7	Schriften	877
19.4	Buttons	878
19.4.1	Die Basisklasse »ButtonBase«	878
19.4.2	Das Steuerelement »Button«	879
19.4.3	Das Steuerelement »ToggleButton«	880
19.4.4	Das Steuerelement »RepeatButton«	881
19.4.5	Das Steuerelement »Checkbox«	882
19.4.6	Das Steuerelement »RadioButton«	882
19.5	Einfache Eingabesteuerelemente	884
19.5.1	Das Steuerelement »Label«	884
19.5.2	Das Steuerelement »TextBox«	885
19.5.3	Das Steuerelement »PasswordBox«	888
19.5.4	Das Steuerelement »TextBlock«	889
19.5.5	Das Steuerelement »ToolTip«	892
19.6	FlowDocuments	893
19.6.1	Allgemeine Beschreibung	893
19.6.2	Eigenschaften eines FlowDocuments	895
19.6.3	Blöcke eines FlowDocuments	895
19.6.4	Inline-Elemente	899
19.6.5	FlowDocuments mit Code erzeugen	902
19.6.6	Speichern und Laden von FlowDocuments	905
19.7	Das Element »FlowDocumentViewer«	905
19.7.1	Das Anzeigeelement »FlowDocumentScrollViewer«	905
19.7.2	Das Anzeigeelement »FlowDocumentPageViewer«	906
19.7.3	Das Anzeigeelement »FlowDocumentReader«	907
19.8	XPS-Dokumente mit »DocumentViewer«	907
19.8.1	Allgemeines zum XPS-Format	907
19.8.2	Beispielprogramm	908
19.8.3	Das Steuerelement »RichTextBox«	909
19.9	WPF-Listenelemente – ItemControls	914
19.9.1	Das Steuerelement »ListBox«	914
19.9.2	ComboBox	917
19.9.3	Das Steuerelement »ListView«	918
19.9.4	Das Steuerelement »TreeView«	921

19.9.5	Das Steuerelement »TabControl«	922
19.9.6	Menüleisten	923
19.9.7	Kontextmenüs	926
19.9.8	Symbolleisten	928
19.9.9	Die Statusleiste	931
19.10	Weitere Steuerelemente	932
19.10.1	Die »ProgressBar«	932
19.10.2	Das Steuerelement »Slider«	933
19.10.3	Das »GroupBox«-Steuerelement	933
19.10.4	Das Steuerelement »ScrollViewer«	935
19.10.5	Das Steuerelement »Expander«	936
19.10.6	Das Steuerelement »Border«	938
19.10.7	Die »Image«-Komponente	939
19.10.8	Grafik zur Laufzeit laden	940

20 Konzepte der WPF

941

20.1	Ressourcen	941
20.1.1	Wo logische Ressourcen definiert werden können	941
20.1.2	Definition logischer Ressourcen	943
20.1.3	Statische und dynamische Ressourcen	946
20.1.4	Ressourcen in Ressourcendateien	948
20.1.5	Suche nach einer Ressource	950
20.1.6	Ressourcen mit C#-Code	951
20.1.7	Abrufen von Systemressourcen	952
20.2	Styles	954
20.2.1	Einfache Stile	954
20.2.2	Typisierte Stile	957
20.2.3	EventSetter	960
20.3	Trigger	962
20.3.1	Eigenschaftstrigger	963
20.3.2	Ereignistrigger	965
20.3.3	Datentrigger	967
20.4	Templates	968
20.4.1	Grundlagen der Templates	968
20.4.2	Verfeinerung des Entwurfs	970
20.4.3	Definition innerhalb eines Styles	973
20.4.4	Den Visual Tree ermitteln	974
20.5	Commands	979
20.5.1	Vordefinierte Commands	979
20.5.2	Beispielanwendung	980
20.5.3	Commando-Ziel festlegen	981

20.5.4	Commands an Ereignisse binden	982
20.5.5	Commands programmieren	983

21 Datenbindung **985**

21.1	Wichtige Klassen	985
21.1.1	Die Klasse »Binding«	985
21.1.2	Die Klasse »DataContext«	986
21.2	Einfache Datenbindung	987
21.2.1	Bindung mit Code erzeugen	988
21.2.2	Bindungsrichtung	989
21.2.3	Änderung der Datenquelle	991
21.2.4	Weitere Möglichkeiten im Zusammenhang mit der Datenbindung	994
21.2.5	Die Ereignisse »SourceUpdated« und »TargetUpdated«	995
21.3	Verschiedene Datenbindungsquellen	996
21.3.1	Bindung an einfache Objekte	996
21.3.2	Auflistungen binden	997
21.4	DataTemplates festlegen	999
21.4.1	Trigger	1001
21.5	Daten konvertieren	1003
21.5.1	Beispielprogramm	1005
21.5.2	Mehrfachbindungen und »Converter«-Klassen	1006
21.6	Die Klasse »ObjectDataProvider«	1008
21.7	WPF und ADO.NET	1010
21.7.1	Ausgabe in einer ListBox	1010
21.7.2	Eine Tabelle im DataGrid-Steuerelement	1013
21.7.3	WPF und LINQ to SQL	1019

22 2D-Grafik **1021**

22.1	Shapes	1021
22.1.1	Allgemeine Beschreibung	1021
22.1.2	Line-Elemente	1022
22.1.3	Ellipse- und Rectangle-Elemente	1022
22.1.4	Polygon- und Polyline-Elemente	1023
22.1.5	Darstellung der Linien	1023
22.2	Path-Elemente	1025
22.2.1	GeometryGroup	1026
22.2.2	CombinedGeometry	1027
22.2.3	PathGeometry	1028

22.3	Brush-Objekte	1029
22.3.1	SolidColorBrush	1030
22.3.2	LinearGradientBrush	1031
22.3.3	RadialGradientBrush	1033
22.3.4	TileBrush	1034
22.3.5	ImageBrush	1036
22.3.6	VisualBrush	1037
22.3.7	DrawingBrush	1039

23 ADO.NET – verbindungsorientierte Objekte **1041**

23.1	Datenprovider	1042
23.2	Verbindung zu einer Datenbank herstellen	1043
23.2.1	Das »Connection«-Objekt	1043
23.2.2	Verbindungszeichenfolge	1044
23.2.3	Verbindung mit einer SQL Server-Instanz aufbauen	1045
23.2.4	Öffnen und Schließen einer Verbindung	1049
23.2.5	Verbindungspooling	1052
23.2.6	Ereignisse eines »Connection«-Objekts	1057
23.2.7	Verbindungszeichenfolgen aus einer Konfigurationsdatei abrufen	1058
23.2.8	Verbindungen mit dem »OleDb«-Datenprovider	1061

24 ADO.NET – Das Command-Objekt **1065**

24.1	Das »SqlCommand«-Objekt	1065
24.1.1	Erzeugen eines »SqlCommand«-Objekts	1066
24.1.2	Die Methode »CreateCommand« des »Connection«-Objekts	1067
24.1.3	Ausführen des »SqlCommand«-Objekts	1067
24.1.4	Die Eigenschaft »CommandTimeout« des »SqlCommand«-Objekts	1068
24.2	Aktionsabfragen absetzen	1068
24.2.1	Datensätze hinzufügen	1068
24.2.2	Datensätze löschen	1069
24.2.3	Datensätze ändern	1070
24.2.4	Abfragen, die genau ein Ergebnis liefern	1070
24.3	Das »SqlDataReader«-Objekt	1070
24.3.1	Datensätze einlesen	1071
24.3.2	Schließen des »SqlDataReader«-Objekts	1074
24.3.3	MARS (Multiple Active Resultsets)	1074
24.3.4	Batch-Abfragen mit »NextResult« durchlaufen	1075
24.3.5	Schema eines »SqlDataReader«-Objekts untersuchen	1076

24.4	Parametrisierte Abfragen	1079
24.4.1	Parametrisierte Abfragen mit dem SqlClient-Datenprovider	1079
24.4.2	Die Klasse »SqlParameter«	1081
24.4.3	Asynchrone Abfragen	1082
24.4.4	Gespeicherte Prozeduren (Stored Procedures).....	1086

25 ADO.NET – Der SqlDataAdapter 1093

25.1	Was ist ein »DataAdapter«?	1093
25.2	Konstruktoren der Klasse »DataAdapter«	1095
25.3	Mit dem »SqlDataAdapter« arbeiten	1096
25.3.1	Die Eigenschaft »SelectCommand«	1096
25.3.2	Lokalen Datenspeicher mit »Fill« füllen	1096
25.3.3	Öffnen und Schließen von Verbindungen	1097
25.3.4	Doppelter Aufruf der »Fill«-Methode	1098
25.3.5	Mehrere »DataAdapter«-Objekte aufrufen	1099
25.3.6	Spalten- und der Tabellenbezeichner einer »DataTable«.....	1099
25.3.7	Paging mit der »Fill«-Methode	1100
25.4	Tabellenzuordnung mit der Klasse »TableMappings«	1100
25.4.1	Spaltenzuordnungen in einem »DataSet«	1102
25.4.2	Spaltenzuordnungen einer »DataTable«	1104
25.4.3	Die Eigenschaft »MissingMappingAction« des »DataAdapter«	1104
25.5	Das Ereignis »FillError« des »SqlDataAdapter«	1105

26 ADO.NET – Daten im lokalen Speicher 1107

26.1	»DataSet«-Objekte verwenden	1108
26.1.1	»DataSet«-Objekte erzeugen	1108
26.1.2	Anatomie einer »DataTable«	1108
26.1.3	Zugriff auf eine Tabelle im »DataSet«	1109
26.1.4	Zugriff auf die Ergebnisliste	1110
26.1.5	Dateninformationen in eine XML-Datei schreiben	1111
26.2	Gültigkeitsprüfung im »DataSet«	1112
26.2.1	Dem »DataSet« Schema-Informationen übergeben	1112
26.2.2	Eigenschaften einer »DataColumn«, die zur Gültigkeitsprüfung dienen	1114
26.2.3	Die »Constraints«-Klassen einer »DataTable«.....	1115
26.2.4	Schema mit Programmcode erzeugen	1116
26.2.5	Schema-Informationen mit »SqlDataAdapter« abrufen	1118
26.3	Änderungen in einer DataTable vornehmen	1121
26.3.1	Editieren einer DataRow	1121
26.3.2	Datenzeile löschen	1123

26.3.3	Neue Datenzeile hinzufügen	1123
26.3.4	Der Sonderfall: Autoinkrementspalten	1124
26.3.5	Was passiert bei der Änderung einer Datenzeile?.....	1126
26.3.6	Manuelles Steuern der Eigenschaft »DataRowState«	1130
26.4	Mit mehreren Tabellen arbeiten	1131
26.4.1	Der Weg über JOIN-Abfragen	1131
26.4.2	Mehrere Tabellen in einem »DataSet«	1133
26.4.3	Eine »DataRelation« erzeugen	1133
26.4.4	»DataRelation« und Einschränkungen	1134
26.4.5	In Beziehung stehende Daten suchen	1136
26.4.6	Ergänzung zum Speichern von Schema-Informationen in einer XML-Schema-Datei	1139
26.5	Filtern und Suchen in einer DataTable	1139
26.5.1	Die Methode »Find«	1139
26.5.2	Die Methode »Select«	1140
26.6	Objekte vom Typ »DataView«	1141
26.6.1	»DataView« erzeugen	1142
26.6.2	Auf die Datenzeilen in einer »DataView« zugreifen	1143
26.6.3	Die Eigenschaft »Sort« und die Methode »Find«.....	1143
26.6.4	Die Methode »FindRows«	1143
26.6.5	Die Eigenschaft »RowFilter«	1144
26.6.6	Die Eigenschaft »RowStateFilter«	1144
26.6.7	Änderungen an einem »DataView«-Objekt	1145
26.6.8	Aus einer »DataView« eine »DataTable« erzeugen.....	1146

27 ADO NET – Aktualisieren der Datenbank **1149**

27.1	Aktualisieren mit dem »CommandBuilder«	1149
27.1.1	Von »SqlCommandBuilder« generierte Aktualisierungsstatements	1151
27.1.2	Konfliktsteuerung in einer Mehrbenutzerumgebung	1152
27.1.3	Die Eigenschaft »ConflictOption« des »SqlCommandBuilder«	1155
27.1.4	Die Eigenschaft »SetAllValues«	1156
27.2	Manuell gesteuerte Aktualisierung	1156
27.2.1	Eigene Aktualisierungslogik	1157
27.2.2	Beispielprogramm	1159
27.3	Konfliktanalyse	1161
27.3.1	Benutzer über fehlgeschlagene Aktualisierungen informieren	1161
27.3.2	Konfliktverursachende Datenzeilen bei der Datenbank abfragen	1163
27.4	Neue Autoinkrementwerte abrufen	1168

28 Stark typisierte DataSets	1171
28.1 Stark typisierte DataSets erzeugen	1171
28.1.1 Typisierte DataSets mit dem Visual Studio Designer erstellen	1171
28.1.2 Das Kommandozeilentool »XSD.exe«	1174
28.2 Anatomie eines typisierten DataSet	1175
28.2.1 Datenzeilen einer Tabelle ausgeben	1176
28.2.2 Datenzeilen hinzufügen	1179
28.2.3 Datenzeilen bearbeiten	1180
28.2.4 Datenzeilen suchen	1180
28.2.5 NULL-Werte im typisierten DataSet	1180
28.2.6 Daten in einem hierarchischen DataSet	1181
28.3 Typisierte DataSets manuell im Designer erzeugen	1182
28.3.1 DataTable manuell erzeugen	1182
28.3.2 Der DataTable Spalten hinzufügen	1182
28.3.3 Beziehungen zwischen den Tabellen erstellen	1183
28.3.4 Weitergehende Betrachtungen	1185
28.4 Der TableAdapter	1185
28.4.1 TableAdapter mit Visual Studio erzeugen	1185
28.4.2 Die Methode »Fill« des TableAdapters	1191
28.4.3 Die Methode »GetData«	1192
28.4.4 Die Methode »Update«	1192
28.4.5 Aktualisieren mit den »DBDirect«-Methoden	1192
28.4.6 TableAdapter mit mehreren Abfragen	1193
28.4.7 Änderungen an einem TableAdapter vornehmen	1196
28.5 Fazit: Typisierte oder nicht typisierte DataSets?	1197
29 LINQ to SQL	1199
29.1 Allgemeine Grundlagen	1199
29.2 Objektzuordnung mit Entitätsklassen	1200
29.3 Mapping von Objekten	1202
29.3.1 Das »Table«-Attribut	1202
29.3.2 Das »Column«-Attribut	1202
29.4 Verknüpfungen zwischen Entitäten	1203
29.4.1 Der Typ »EntityRef<T>«	1205
29.4.2 Verzögertes Laden	1206
29.4.3 Der Typ »EntitySet<T>«	1206
29.4.4 Ein weiteres Beispiel	1208
29.4.5 Sofortiges Laden der Daten	1208

29.5	Tools zur Erzeugung von Entitätsklassen	1210
29.6	Die Klasse »DataContext«	1211
29.6.1	Verbindungsaufbau	1211
29.6.2	Daten abfragen	1211
29.6.3	Von einer LINQ-Abfrage erzeugtes SQL-Statement ausgeben	1212
29.6.4	Aktualisieren der Daten	1213
29.6.5	Konflikte behandeln	1216
29.7	Der LINQ to SQL-Designer (O/R-Designer)	1222
29.7.1	Handhabung des O/R-Designers	1222
29.7.2	Die abgeleitete »DataContext«-Klasse	1226
29.7.3	Entitätsklassen	1229
30 Weitergabe von Anwendungen		1231
30.1	Weitergabe mit MS-Installer	1231
30.1.1	Weitergabeprojekte	1232
30.1.2	Der Windows Installer	1233
30.1.3	Weitergabeprojekte mit dem Setup-Assistenten	1233
30.1.4	Editoren eines Weitergabeprojekts	1236
30.1.5	Der Dateisystem-Editor	1238
30.1.6	Der Registrierungs-Editor	1245
30.1.7	Der Dateityp-Editor	1247
30.1.8	Der Benutzeroberflächen-Editor	1250
30.1.9	Editor für benutzerdefinierte Aktionen	1261
30.1.10	Editor für Startbedingungen	1262
30.2	ClickOnce-Verteilung	1267
30.2.1	Allgemeine Beschreibung	1267
30.2.2	Erstellen einer ClickOnce-Anwendung	1269
30.2.3	Installation einer ClickOnce-Anwendung	1273
Index		1275