

Inhalt

Vorwort	27
----------------------	-----------

Teil I: Einführung in die C#-Programmierung

1 Einführung in die .NET-Technologie	33
1.1 Zur Historie von .NET	34
1.1.1 Warum eine neue Programmiersprache?	34
1.1.2 Wie funktioniert eine .NET-Sprache?	35
1.2 Das .NET-Konzept	36
1.2.1 Das .NET-Framework	36
1.2.2 Die Common Language Specification (CLS)	37
1.2.3 Das Common Type System (CTS)	38
1.2.4 Die Common Language Runtime (CLR)	38
1.2.5 Die .NET-Klassenbibliothek	39
1.2.6 Das Konzept der Namespaces	39
1.2.7 Zugriff auf COM-Komponenten	40
1.2.8 Was sind Assemblies?	41
1.2.9 Metadaten und Manifest	41
1.2.10 Der Reflection-Mechanismus	42
1.2.11 Attribute	42
1.2.12 Serialisierung	43
1.2.13 Multithreading	43

1.3	Der Einstieg in Visual Studio	44
1.3.1	Ein erster Überblick	44
1.3.2	Der Projektmappen-Explorer	46
1.3.3	Der Designer	47
1.3.4	Das Eigenschaften-Fenster	48
1.3.5	Das Codefenster	49
2	Visual C#-Crashkurs für Anfänger	51
2.1	Es lebe der gute alte PAP!	52
2.2	Ab ins Regal!	56
2.3	Aufhören oder weitermachen?	58
2.4	Jedes Programm lässt sich verbessern!	61
2.5	Schluss mit langweiliger Konsole!	64
2.6	Kann es auch etwas anspruchsvoller sein?	68
2.7	Hilfe, mein Programm liefert falsche Ergebnisse!	71
3	Die Sprache C# im Überblick	75
3.1	Grundlagen der C#-Syntax	76
3.1.1	Anweisungen	76
3.1.2	Bezeichner	76
3.1.3	Schlüsselwörter	77
3.1.4	Kommentare	78
3.2	Datentypen und Variablen	79
3.2.1	Fundamentale Typen	79
3.2.2	Namensgebung von Variablen	80
3.2.3	Deklaration von Variablen	80
3.2.4	Initialisierte Variablen	81
3.2.5	Konstanten	81
3.2.6	Typsuffixe	81
3.3	Zeichen und Zeichenketten	82
3.3.1	Der char-Typ	82
3.3.2	Der string-Typ	83
3.4	Weitere wichtige Datentypen und -features	83
3.4.1	Der object-Datentyp	83
3.4.2	Nullable Types	84
3.4.3	Typinferenz	85
3.4.4	Gültigkeitsbereiche und Sichtbarkeit	86

3.5	Konvertieren von Datentypen	87
3.5.1	Implizite und explizite Konvertierung	87
3.5.2	Welcher Datentyp passt zu welchem?	88
3.5.3	Konvertieren von <code>bool</code>	89
3.5.4	Konvertieren von <code>string</code>	89
3.5.5	Boxing und Unboxing	91
3.6	Operatoren	93
3.6.1	Arithmetische Operatoren	94
3.6.2	Zuweisungsoperatoren	96
3.6.3	Logische Operatoren	96
3.6.4	Rangfolge der Operatoren	99
3.7	Kontrollstrukturen	100
3.7.1	Verzweigungsbefehle	100
3.7.2	Programmschleifen	103
3.8	Benutzerdefinierte Datentypen	106
3.8.1	Enumerationen	106
3.8.2	Strukturen	107
3.9	Nutzerdefinierte Methoden	109
3.9.1	Methoden mit Rückgabewert	110
3.9.2	Methoden ohne Rückgabewert	111
3.9.3	Parameterübergabe mit <code>ref</code>	112
3.9.4	Parameterübergabe mit <code>out</code>	113
3.9.5	Überladene Methoden	114
4	Objektorientiertes Programmieren	115
4.1	Strukturierte versus objektorientierte Programmierung	116
4.1.1	Was bedeutet strukturiert?	116
4.1.2	Was heißt objektorientiert?	117
4.2	Grundbegriffe der OOP	118
4.2.1	Objekt, Klasse, Instanz	118
4.2.2	Kapselung und Wiederverwendbarkeit	118
4.2.3	Vererbung und Polymorphie	119
4.2.4	Sichtbarkeit von Klassen und ihren Mitgliedern	120
4.2.5	Allgemeiner Aufbau einer Klasse	121
4.3	Ein Objekt erzeugen	122
4.3.1	Referenzieren und Instanziieren	122
4.3.2	Klassische Initialisierung	123
4.3.3	Objekt-Initialisierer	124

4.3.4	Arbeiten mit dem Objekt	124
4.3.5	Zerstören des Objekts	124
4.4	OOP-Einführung am Beispiel	125
4.4.1	Vorbereitungen	125
4.4.2	Klasse definieren	126
4.4.3	Bemerkungen	126
4.4.4	Objekt erzeugen und initialisieren	127
4.4.5	Objekt verwenden	127
4.4.6	Intellisense – die hilfreiche Fee	127
4.4.7	Objekt testen	128
4.4.8	Bemerkungen	128
4.5	Eigenschaften	129
4.5.1	Eigenschaften kapseln	129
4.5.2	Eigenschaften mit Zugriffsmethoden berechnen	131
4.5.3	Lese-/Schreibschutz für Eigenschaften	133
4.5.4	Statische Eigenschaften	133
4.6	Methoden	134
4.6.1	Öffentliche und private Methoden	134
4.6.2	Statische Methoden	135
4.7	Ereignisse	137
4.7.1	Ereignis-Delegate hinzufügen	137
4.7.2	Ereignisse verwenden	140
4.8	Konstruktor und Destruktor	143
4.8.1	Der Konstruktor erzeugt das Objekt	144
4.8.2	Destruktor und Garbage Collector räumen auf	145
4.9	Vererbung und Polymorphie	148
4.9.1	Vererbungsbeziehungen im Klassendiagramm	148
4.9.2	Methoden überschreiben	149
4.9.3	Klassen implementieren	149
4.9.4	Objekte implementieren	153
4.9.5	Vielgestaltige Methoden	154
4.9.6	Allgemeine Hinweise und Regeln zur Vererbung	157
4.10	Besondere Klassen	158
4.10.1	Abstrakte Klassen	158
4.10.2	Versiegelte Klassen	160
4.10.3	Partielle Klassen	160
4.10.4	Die Basisklasse System.Object	161

5	Weitere C#-Sprachfeatures	163
5.1	Namespaces (Namensräume)	164
5.1.1	Ein kleiner Überblick	164
5.1.2	Einen eigenen Namespace einrichten	165
5.1.3	using-Anweisung	166
5.1.4	Namespace-Alias	167
5.1.5	Namespace Alias Qualifizierer	167
5.2	Datenfelder (Arrays)	168
5.2.1	Ein Array deklarieren	169
5.2.2	Ein Array instanziieren	169
5.2.3	Ein Array initialisieren	169
5.2.4	Der Zugriff auf Array-Elemente	170
5.2.5	Zugriff mittels Schleife	171
5.2.6	Mehrdimensionale Arrays	172
5.2.7	Zuweisen von Arrays	174
5.2.8	Arrays mit Strukturvariablen	175
5.2.9	Löschen von Arrays	176
5.2.10	Eigenschaften und Methoden der Array-Klasse	177
5.2.11	Übergabe von Arrays	178
5.2.12	Indexer	179
5.3	Zeichenkettenverarbeitung	181
5.3.1	Strings zuweisen	182
5.3.2	Eigenschaften und Methoden von Strings	182
5.3.3	Strings in Char-Arrays verwandeln	184
5.3.4	Wichtige Methoden der String-Klasse	185
5.4	Zahlen formatieren	187
5.4.1	Die ToString-Methode	187
5.4.2	Die Format-Methode	189
5.5	Vordefinierte Methoden	190
5.5.1	Mathematische Funktionen	190
5.5.2	Datums- und Zeitfunktionen	193
5.6	Überladen von Operatoren	195
5.6.1	Syntaxregeln	195
5.6.2	Addition von Arrays	195
5.7	Auflistungen (Collections)	197
5.7.1	Beziehungen zwischen den Schnittstellen	197
5.7.2	IEnumerable	197
5.7.3	ICollection	198

5.7.4	IList	199
5.7.5	Die ArrayList-Collection	199
5.7.6	Die Hashtable	200
5.8	Generische Datentypen	201
5.8.1	Wie es früher war	202
5.8.2	Typsicherheit durch Generics	203
5.8.3	List-Collection ersetzt ArrayList	204
5.8.4	Über die Vorteile generischer Collections	205
5.8.5	Typbeschränkungen mit Constraints	206
5.8.6	Generische Methoden	206
5.8.7	Iteratoren	207
5.9	Was sind Delegates?	207
5.9.1	Delegates sind Methodenzeiger	208
5.9.2	Delegate-Typ deklarieren	208
5.9.3	Delegate-Objekt erzeugen	208
5.9.4	Delegates vereinfacht instanziieren	210
5.9.5	Anonyme Methoden	210
5.9.6	Lambda-Ausdrücke	212
6	Einführung in LINQ	215
6.1	LINQ-Grundlagen	216
6.1.1	Die LINQ-Architektur	216
6.1.2	LINQ-Implementierungen	216
6.1.3	Anonyme Typen	217
6.1.4	Erweiterungsmethoden	218
6.2	Abfragen mit LINQ	219
6.2.1	Grundlegendes zur LINQ-Syntax	220
6.2.2	Die Projektionsoperatoren Select und SelectMany	223
6.2.3	Der Restriktionsoperator Where	225
6.2.4	Die Sortierungsoperatoren OrderBy und ThenBy	225
6.2.5	Der Gruppierungsoperator GroupBy	227
6.2.6	Verknüpfungen mit Join	229
6.2.7	Aggregat-Operatoren	230
6.2.8	Verzögertes Ausführen von Abfragen	231
6.2.9	Konvertierungsmethoden	232

Teil II: Bausteine komplexer Anwendungen

7 Kommandozeilenanwendungen	235
7.1 Grundaufbau/Konzepte	236
7.1.1 Unser Hauptprogramm – Program.cs	237
7.1.2 Rückgabe eines Fehlerstatus	238
7.1.3 Parameterübergabe	239
7.1.4 Zugriff auf die Umgebungsvariablen	240
7.2 Die Kommandozentrale: System.Console	241
7.2.1 Eigenschaften	241
7.2.2 Methoden/Ereignisse	242
7.2.3 Textausgaben	242
7.2.4 Farbangaben	243
7.2.5 Tastaturabfragen	245
7.2.6 Arbeiten mit Streamdaten	246
7.2.7 DEMO: Farbige Konsolenanwendung	247
7.2.8 Weitere nützliche Informationen	249
8 Einführung in Windows Forms-Anwendungen	251
8.1 Grundaufbau/Konzepte	252
8.1.1 Das Hauptprogramm – Program.cs	253
8.1.2 Die Oberflächendefinition – Form1.Designer.cs	256
8.1.3 Die Spielwiese des Programmierers – Form1.cs	257
8.1.4 Die Datei AssemblyInfo.cs	258
8.1.5 Resources.resx/Resources.Designer.cs	259
8.1.6 Settings.settings/Settings.Designer.cs	260
8.1.7 Settings.cs	261
8.2 Ein Blick auf die Application-Klasse	262
8.2.1 Eigenschaften	263
8.2.2 Methoden	264
8.2.3 Ereignisse	265
8.3 Allgemeine Eigenschaften von Komponenten	266
8.3.1 Übersicht	266
8.3.2 Font	267
8.3.3 Handle	268
8.3.4 Tag	269
8.3.5 Modifiers	269

8.4	Allgemeine Ereignisse von Komponenten	270
8.4.1	Die Eventhandler-Argumente	270
8.4.2	Das sender-Objekt	270
8.4.3	Der Parameter e	271
8.4.4	Mausereignisse	272
8.4.5	Tastaturereignisse	273
8.4.6	Wozu brauchen wir KeyPreview?	274
8.4.7	Weitere Ereignisse	275
8.4.8	Validitätsprüfungen	275
8.5	Allgemeine Methoden von Komponenten	276
9	Windows-Formulare verwenden	279
9.1	Übersicht zur Form-Klasse	280
9.1.1	Wichtige Eigenschaften	280
9.1.2	Wichtige Methoden	282
9.1.3	Wichtige Ereignisse	283
9.2	Praktische Aufgabenstellungen	284
9.2.1	Ein Fenster anzeigen	284
9.2.2	Einen Splash Screen beim Anwendungsstart anzeigen	287
9.2.3	Eine Sicherheitsabfrage vor dem Schließen anzeigen	289
9.2.4	Ein Formular durchsichtig machen	290
9.2.5	Die Tabulatorreihenfolge festlegen	290
9.2.6	Ausrichten und Platzieren von Komponenten im Formular	291
9.2.7	Spezielle Panels für flexibles Layout	294
9.2.8	Menüs erzeugen	295
9.3	MDI-Anwendungen	298
9.3.1	Unsere Verwaltungszentrale: Das MDI-Hauptfenster	298
9.3.2	Die Kindfenster	299
9.3.3	Automatisches Anordnen der Kindfenster	300
9.3.4	Zugriff auf die geöffneten MDI-Kindfenster	301
9.3.5	Zugriff auf das aktive MDI-Kindfenster	302
9.3.6	Kombinieren der Kindfenstermenüs mit dem MDIContainer	302
10	Wichtige Windows Forms-Komponenten	303
10.1	Grundlegende Techniken	304
10.1.1	Hinzufügen von Komponenten	304
10.1.2	Komponenten zur Laufzeit erzeugen	305

10.2	Allgemeine Steuerelemente	306
10.2.1	Label	306
10.2.2	LinkLabel	307
10.2.3	Button	308
10.2.4	TextBox	309
10.2.5	MaskedTextBox	312
10.2.6	CheckBox	313
10.2.7	RadioButton	314
10.2.8	ListBox	315
10.2.9	CheckedListBox	316
10.2.10	ComboBox	317
10.2.11	PictureBox	318
10.2.12	DateTimePicker	318
10.2.13	MonthCalendar	319
10.2.14	HScrollBar, VScrollBar	319
10.2.15	TrackBar	320
10.2.16	NumericUpDown	320
10.2.17	DomainUpDown	321
10.2.18	ProgressBar	322
10.2.19	RichTextBox	322
10.2.20	ListView	323
10.2.21	TreeView	329
10.2.22	WebBrowser	333
10.3	Container	334
10.3.1	FlowLayout/TableLayout/SplitContainer	334
10.3.2	Panel	335
10.3.3	GroupBox	335
10.3.4	TabControl	336
10.3.5	ImageList	338
10.4	Menüs und Symbolleisten	339
10.4.1	MenuStrip und ContextMenuStrip	339
10.4.2	ToolStrip	339
10.4.3	StatusStrip	340
10.4.4	ToolStripContainer	340
11	Grafikausgabe und Drucken	341
11.1	Grundlegende Konzepte	342
11.1.1	Die Grafikzentrale: GDI+	342

11.1.2	Wo finde ich was?	343
11.2	Grafikanzeige und -manipulation	344
11.2.1	Grafikanzeige mit der PictureBox	345
11.2.2	Die Image-Klasse	346
11.2.3	Grafiken zur Laufzeit zuweisen	347
11.2.4	Grafiken in verschiedenen Formaten sichern	347
11.2.5	Wichtige Grafikeigenschaften ermitteln	348
11.2.6	Erzeugen von Vorschaugrafiken (Thumbnails)	348
11.2.7	Grafiken drehen	349
11.2.8	Skalieren von Grafiken	349
11.3	Koordinatensystem von GDI+	350
11.3.1	Grundsätzlicher Aufbau	350
11.3.2	Globale Koordinaten	351
11.3.3	Seitenkoordinaten (globale Transformation)	351
11.3.4	Gerätekordinaten (Seitentransformation)	353
11.4	Wichtige Grafikmethoden	354
11.4.1	Das zentrale Graphics-Objekt	354
11.4.2	Punkte zeichnen/abfragen	355
11.4.3	Linien	356
11.4.4	Kantenglättung mit Antialiasing	357
11.4.5	PolyLine	358
11.4.6	Rechtecke	358
11.4.7	Polygone	359
11.4.8	Splines	360
11.4.9	Bézierkurven	361
11.4.10	Kreise und Ellipsen	362
11.4.11	Tortenstück (Segment)	362
11.4.12	Bogenstück	364
11.4.13	Textausgabe	365
11.4.14	Ausgabe von Grafiken	368
11.5	Die wichtigsten Grafikobjekte	369
11.5.1	Einfache Objekte	369
11.5.2	Vordefinierte Objekte	370
11.5.3	Farben/Transparenz	372
11.5.4	Stifte (Pen)	373
11.5.5	Pinsel (Brush)	375
11.5.6	SolidBrush	375
11.5.7	HatchBrush	375

11.5.8	TextureBrush	376
11.5.9	LinearGradientBrush	377
11.5.10	PathGradientBrush	378
11.5.11	Fonts	379
11.6	Die Grafik-Standarddialoge	380
11.6.1	FontDialog für die Schriftauswahl	380
11.6.2	Farbauswahl mit ColorDialog	381
11.7	Druckausgabe/Druckvorschau	383
11.7.1	Grundkonzept	383
11.7.2	Programmiermodell	384
11.7.3	Komponenten für die Druckausgabe	385
11.7.4	Praktische Aufgabenstellungen	386
11.7.5	Die Druckdialoge	388
11.7.6	Ein eigenes Druckvorschau-Fenster	392
11.7.7	DEMO: Komplette Druckausgabe	393
12	Zugriff auf das Dateisystem	403
12.1	Grundlagen	404
12.1.1	Das Datei-System von Windows	404
12.1.2	Klassen für Verzeichnis- und Dateioperationen	404
12.1.3	Statische versus Instanzen-Klassen	405
12.2	Operationen auf Verzeichnisebene	406
12.2.1	Verzeichnisse erzeugen und löschen	406
12.2.2	Verzeichnisse umbenennen und verschieben	407
12.2.3	Aktuelles Verzeichnis bestimmen	407
12.2.4	Unterverzeichnisse ermitteln	407
12.2.5	Alle Laufwerke ermitteln	408
12.2.6	Im Verzeichnis enthaltene Dateien ermitteln	409
12.2.7	Dateien kopieren und verschieben	409
12.2.8	Dateien umbenennen	410
12.2.9	Dateiattribute feststellen	410
12.2.10	Die FileAttribute-Enumeration	411
12.3	Mehr zur FileInfo-Klasse	412
12.3.1	Weitere wichtige Eigenschaften	412
12.3.2	GetFileSystemInfos-Methode	412
12.4	Zugriffsberechtigungen	413
12.4.1	ACL und ACE	413
12.4.2	SetAccessControl-Methode	413

12.4.3	Zugriffsrechte anzeigen	414
12.5	Weitere wichtige Klassen	415
12.5.1	Die Path-Klasse	415
12.5.2	Die Klasse FileSystemWatcher	415
13	Dateien lesen und schreiben	417
13.1	Das Grundprinzip der Datenpersistenz	418
13.1.1	Dateien und Streams	418
13.1.2	Die wichtigsten Klassen	418
13.1.3	Erzeugen eines Streams	419
13.2	Dateiparameter	419
13.2.1	Die FileAccess-Enumeration	420
13.2.2	Die FileMode-Enumeration	420
13.2.3	Die FileShare-Enumeration	420
13.3	Textdateien	421
13.3.1	Eine Textdatei beschreiben bzw. neu anlegen	421
13.3.2	Eine Textdatei lesen	422
13.4	Binärdateien	422
13.4.1	Lese-/Schreibzugriff	422
13.4.2	Die Methoden ReadAllBytes/WriteAllBytes	423
13.4.3	Varianten zum Erzeugen von BinaryReader/BinaryWriter	423
13.5	Sequenzielle Dateien	424
13.5.1	Lesen und schreiben von strukturierten Daten	424
13.5.2	Serialisieren von Objekten	425
13.6	Dateien verschlüsseln und komprimieren	426
13.6.1	Das Methodenpärchen Encrypt-/Decrypt	426
13.6.2	Verschlüsseln unter Windows XP/Vista	426
13.6.3	Verschlüsseln mittels der CryptoStream-Klasse	427
13.6.4	Dateien komprimieren	428
13.7	Dateidialoge	429
13.7.1	Anzeige und Auswertung	429
13.7.2	Wichtige Eigenschaften	430
13.7.3	Verwenden von Dateifiltern	431
14	Einführung in ADO.NET	433
14.1	Eine Übersicht	434
14.1.1	Die ADO.NET-Klassenhierarchie	434
14.1.2	Die Klassen der Datenprovider	435

14.1.3	Die Zusammenhänge zwischen den ADO.NET-Klassen	437
14.1.4	DEMO: ADO.NET-Objekte im Einsatz	438
14.2	Das Connection-Objekt	440
14.2.1	Allgemeiner Aufbau	440
14.2.2	OleDbConnection	440
14.2.3	Schließen einer Verbindung	441
14.2.4	Eigenschaften des Connection-Objekts	442
14.2.5	Methoden des Connection-Objekts	444
14.2.6	Der ConnectionStringBuilder	445
14.3	Das Command-Objekt	445
14.3.1	Erzeugen und Anwenden eines Command-Objekts	445
14.3.2	Erzeugen mittels CreateCommand-Methode	446
14.3.3	Eigenschaften des Command-Objekts	446
14.3.4	DEMO: Eine Auswahlabfrage absetzen	448
14.3.5	Methoden des Command-Objekts	451
14.3.6	DEMO: Eine Aktionsabfrage ausführen	452
14.4	Parameter-Objekte	454
14.4.1	Erzeugen und Anwenden eines Parameter-Objekts	454
14.4.2	Eigenschaften des Parameter-Objekts	455
14.5	Das CommandBuilder-Objekt	456
14.5.1	Erzeugen	456
14.5.2	Anwenden	456
14.6	Das DataReader-Objekt	457
14.6.1	DataReader erzeugen	457
14.6.2	Daten lesen	458
14.6.3	Eigenschaften DataReaders	459
14.6.4	Methoden des DataReaders	459
14.7	Das DataAdapter-Objekt	459
14.7.1	DataAdapter erzeugen	460
14.7.2	Command-Eigenschaften	461
14.7.3	Fill-Methode	461
14.7.4	Update-Methode	462
14.7.5	DEMO: Die Datenbank aktualisieren	463
15	Das DataSet	467
15.1	Grundlagen	467
15.1.1	Die wichtigsten Klassen	468
15.1.2	Die Objekthierarchie	468

15.1.3	Erzeugen eines DataSets	469
15.2	Die DataTable	471
15.2.1	DataTable erzeugen	471
15.2.2	Spalten hinzufügen	472
15.2.3	Zeilen zur DataTable hinzufügen	472
15.2.4	Zugriff auf den Inhalt einer DataTable	473
15.3	Die DataView	475
15.3.1	Erzeugen einer DataView	475
15.3.2	Sortieren und Filtern von Datensätzen	476
15.3.3	DEMO: Im DataView sortieren/filtern	476
15.3.4	Suche von Datensätzen	478
15.3.5	DEMO: Datensätze suchen	478
15.4	Typisierte DataSets	480
15.4.1	Was ist ein typisiertes DataSet?	480
15.4.2	Das Konzept der Datenquellen	482
15.4.3	Typisierte DataSets und TableAdapter	482
16	Datenbindung unter ADO.NET	485
16.1	Allgemeine Varianten der Datenbindung	486
16.1.1	Manuelle Datenbindung an einfache Datenfelder	486
16.1.2	Manuelle Datenbindung an Listen und Tabelleninhalte	488
16.1.3	DEMO: Master-Detailbeziehungen im DataGrid	489
16.2	Spezielle Datenbindungen	491
16.2.1	Entwurfszeit-Datenbindung an ein typisiertes DataSet	491
16.2.2	Drag & Drop-Datenbindung	492
16.2.3	Navigieren im DataSet	492
16.2.4	Die Anzeige formatieren	494
16.2.5	DEMO: Arbeiten mit einer Datenquelle	495
Teil III: Methoden und Werkzeuge der Softwareentwicklung		
17	Fehlersuche/-Fehlerbehandlung	501
17.1	Fehlersuche	501
17.1.1	Der Debugger	502
17.1.2	Einzelschritt-Modus	505
17.1.3	Prozedurschritt-Modus	506
17.1.4	DEMO: Wichtige Debugging-Möglichkeiten	506
17.1.5	Das Debug-Objekt	510

17.2	Fehlerbehandlung	511
17.2.1	Fehlermöglichkeiten	511
17.2.2	Der try-catch-Block	511
17.2.3	Der try-finally-Block	516
17.2.4	Das Standardverhalten bei Ausnahmen festlegen	518
17.2.5	Die Exception-Klasse	519
17.2.6	Fehler/Ausnahmen auslösen	520
17.2.7	Eigene Fehlerklassen entwickeln	520
17.2.8	Exceptions zur Entwurfszeit behandeln	522
18	Testprojekte	523
18.1	Testgetriebene Entwicklung (TDD)	524
18.1.1	Konventionelle Vorgehensweise	524
18.1.2	Testgetriebene Entwicklung mit Unit-Tests	524
18.1.3	Unit-Tests unter Visual Studio	525
18.2	Einfache Tests	525
18.2.1	DEMO: Test einer Klasse CKugel	526
18.2.2	Eigene Testmethoden hinzufügen	531
18.3	Datengetriebene Tests (DDT)	532
18.3.1	Das DDT-Prinzip	533
18.3.2	DEMO: Test von Spesenberechnungen	533
18.4	Begriffe und Ergänzungen	537
18.4.1	Behauptungen (Asserts)	537
18.4.2	Der Testkontext	538
18.4.3	Zusätzliche Testattribute	539
18.5	Schlussbemerkungen	540
18.5.1	Nutzen von Unit-Tests	540
18.5.2	Grenzen von Unit-Tests	540
19	Arbeiten mit dem Klassendesigner	541
19.1	Der Klassendesigner im Überblick	542
19.1.1	Wie komme ich zu einem Klassendiagramm?	542
19.1.2	Ein Blick in die Toolbox	543
19.2	Die Bausteine des Klassen-Designers	544
19.2.1	Enumeration	544
19.2.2	Klasse	546
19.2.3	Struktur	548
19.2.4	Abstrakte Klasse	549

19.2.5	Schnittstelle	550
19.2.6	Delegate	552
19.2.7	Zuordnung	554
19.2.8	Vererbung anzeigen	554
19.2.9	Die Anzeige der Diagramme anpassen	554
19.2.10	DEMO: Modellierung des Bestellsystems einer Firma	555
19.3	Mehr zum Klassen-Designer	567
19.3.1	Arbeiten mit dem Objekt-Testcenter	567
19.3.2	Wann lohnt sich der Einsatz des Klassen-Designers?	569
20	Das Microsoft Event Pattern	571
20.1	Das Subjekt-Observer-Entwurfsmuster	572
20.1.1	Warum ist das Observer-Pattern besonders wichtig?	572
20.1.2	Grundprinzip des Observer-Pattern	572
20.2	DEMO: Personalverwaltung	573
20.2.1	Die Datenstruktur	573
20.2.2	Die Ereignisdefinition	575
20.2.3	Die Ereignislogik	577
20.2.4	Das User-Interface	578
20.2.5	Die Datenpersistenz	581
20.2.6	Die Verwaltung des Pattern	583
20.2.7	Test und Diskussion	585
20.2.8	Ein zweiter Observer	586
20.2.9	Zusammenfassung	587
21	Webdienst in Microkernel-Architektur	589
21.1	Grundkonzepte von Webdiensten	590
21.1.1	Zur Bedeutung von Webdiensten	590
21.1.2	Kommunikation per SOAP	590
21.1.3	Die Sprache WSDL	591
21.1.4	Das Funktionsprinzip von Webdiensten	591
21.1.5	Anforderungen an eine Webklasse	593
21.1.6	Schnittstellen-Kompatibilität	593
21.2	Grundkonzept unseres Webdienstes	594
21.2.1	Datenaustausch	594
21.2.2	Datenbankstruktur	595

21.3	Klassische Realisierung	596
21.3.1	Klassendiagramm	596
21.3.2	Konfigurationsdaten	597
21.4	Realisierung in Microkernel-Architektur	598
21.4.1	UML-Klassendiagramm	598
21.4.2	Sequenzdiagramm	599
21.4.3	Die Klasse BookService als Microkernel	600
21.4.4	Die Klasse CCommon	601
21.4.5	Die internen Server	602
21.4.6	Test	607
21.5	Webdienst-Client	608
21.5.1	Das User-Interface	608
21.5.2	Die Programmierung	609
21.5.3	Test	616
21.5.4	Fehlerbeseitigung	618
21.5.5	Schlussbemerkungen	618
22	Verwenden von Ressourcen	619
22.1	Manifestressourcen	620
22.1.1	Erstellen von Manifestressourcen	620
22.1.2	Zugriff auf Manifestressourcen	621
22.2	Typisierte Ressourcen	623
22.2.1	Erzeugen von .resources-Dateien	623
22.2.2	Hinzufügen der .resources-Datei zum Projekt	624
22.2.3	Zugriff auf die Inhalte von .resources-Dateien	624
22.2.4	ResourceManager direkt aus der .resources-Datei erzeugen	625
22.2.5	Was sind .resx-Dateien?	625
22.3	Streng typisierte Ressourcen	626
22.3.1	Erzeugen streng typisierter Ressourcen	626
22.3.2	Verwenden streng typisierter Ressourcen	626
22.3.3	Streng typisierte Ressourcen per Reflection auslesen	627
22.4	Lokalisierte Anwendungen	629
22.4.1	Localizable und Language	629
22.4.2	DEMO: Landesfahnen	629
22.4.3	Einstellen der aktuellen Kultur zur Laufzeit	632

23 Verteilen von Anwendungen	635
23.1 ClickOnce-Deployment	636
23.1.1 Übersicht/Einschränkungen	636
23.1.2 Die Vorgehensweise	637
23.1.3 Ort der Veröffentlichung	637
23.1.4 Anwendungsdateien	638
23.1.5 Erforderliche Komponenten	638
23.1.6 Aktualisierungen	639
23.1.7 Veröffentlichen	640
23.1.8 Verzeichnisstruktur	640
23.1.9 Der Webpublishing-Assistent	642
23.1.10 Neue Versionen erstellen	642
23.2 Setup-Projekte	643
23.2.1 Ein neues Setup-Projekt	643
23.2.2 Dateisystem-Editor	645
23.2.3 Ein erster Test	646
23.2.4 Registrierungs-Editor	647
23.2.5 Dateityp-Editor	648
23.2.6 Benutzeroberflächen-Editor	649
23.2.7 Editor für Startbedingungen	651
23.2.8 Finaler Test	652

Teil IV: Rezepte/Lösungen

24 Wie kann ich ... (Sprache/OOP)	657
24.1 ... Anwendungen von Visual Basic nach C# portieren?	657
24.1.1 Die augenfälligsten Unterschiede	658
24.1.2 Datentypen	659
24.1.3 Operatoren	660
24.1.4 Verzweigungen	661
24.1.5 Schleifen	661
24.1.6 Arrays	662
24.1.7 Strukturen	662
24.1.8 Enumerationen	662
24.1.9 Funktionen, Prozeduren, Methoden	663
24.1.10 Klassendefinition	663
24.1.11 Erzeugen eines Objekts	664

24.1.12 Ereignis definieren und auslösen	664
24.1.13 Ereignis mit Eventhandler verbinden	664
24.2 ... einen String in ein Array kopieren?	665
24.3 ... ein Byte-Array in einen String konvertieren?	667
24.4 ... Strukturvariablen in Arrays einsetzen?	668
24.5 ... eine einzelne Spalte aus einer Matrix kopieren?	671
24.6 ... in einer ArrayList suchen und sortieren?	672
24.7 ... in einer generischen Liste suchen und sortieren?	674
24.8 ... mit Bubblesort sortieren?	676
24.9 ... Zufallszahlen erzeugen?	678
24.10 ... Iterationen verstehen?	679
24.11 ... den Goldenen Schnitt ermitteln?	682
24.12 ... Funktionen rekursiv aufrufen?	684
24.13 ... Zeitmessungen durchführen?	685
24.14 ... Strings vergleichen?	689
24.15 ... Datumsdifferenzen ermitteln?	691
24.16 ... das Alter in Jahren bestimmen?	694
24.17 ... die Monatsdifferenz berechnen?	696
24.18 ... das Datum beweglicher Feiertage berechnen?	697
24.19 ... ersten und letzten Wochentag eines Monats bestimmen?	699
24.20 ... Abschreibungen auf Monatsbasis berechnen?	700
24.21 ... Geldbeträge kaufmännisch runden?	704
24.22 ... Fehler bei mathematischen Operationen behandeln?	706
24.23 ... mit Potenzen und Wurzeln rechnen?	709
24.24 ... überladene/überschriebene Methoden unterscheiden?	710
24.25 ... Delegates verstehen?	713
24.26 ... LINQ-Abfragen verstehen?	715
24.27 ... Eigenschaften sinnvoll kapseln?	718
24.28 ... Aggregation und Vererbung unterscheiden?	721
24.29 ... Referenz-/Wertetypen als Parameter übergeben?	727
25 Wie kann ich ... (Oberfläche/Komponenten)	731
25.1 ... den Inhalt des UI sichern?	731
25.2 ... die Anzeige löschen?	735
25.3 ... die Maus abfragen?	736
25.4 ... Dezimalkomma in Dezimalpunkt umwandeln?	737
25.5 ... mit der TextBox arbeiten?	738

25.6 ... in einer TextBox suchen?	740
25.7 ... die ListBox kennen lernen?	742
25.8 ... RadioButtons und CheckBoxen einsetzen?	745
25.9 ... Objekte in ListBox/ComboBox anzeigen?	747
25.10 ... zur Laufzeit ein Steuerelement erzeugen?	749
25.11 ... Eingaben validieren?	751
25.12 ... ein Graphics-Objekt erzeugen?	754
25.12.1 Variante 1: Verwendung des Paint-Events	754
25.12.2 Variante 2: Überschreiben der OnPaint-Methode	755
25.12.3 Variante 3: Graphics-Objekt mit CreateGraphics erzeugen	756
25.12.4 Variante 4: Graphics-Objekts einer PictureBox nutzen	756
25.13 ... Texte gedreht ausgeben?	757
25.14 ... einen Markierungsrahmen erzeugen?	758
25.15 ... mit Drag & Drop arbeiten?	760
25.16 ... eine Komponente zur Farbauswahl entwickeln?	764
26 Wie kann ich ... (Sonstiges)	769
26.1 ... die Zwischenablage verwenden?	769
26.2 ... mittels Reflection Typinformationen sammeln?	772
26.3 ... den mehrfachen Anwendungsstart verhindern?	775
26.4 ... eine Pause realisieren?	777
26.5 ... Systemtöne und WAV-Dateien wiedergeben?	779
26.6 ... diverse Systeminformationen ermitteln?	781
26.6.1 Betriebssystem (Name, Version, Bootmode)	782
26.6.2 Schriftarten/-Informationen	783
26.6.3 Bildschirme	784
26.6.4 Environment Variablen auslesen	785
26.6.5 Netzwerk (User-Name, PC-Name ...)	785
26.6.6 Hardware-Informationen	786
26.6.7 Energiestatus	787
26.6.8 Anwendung (Pfad, Name, Assembly)	787
26.6.9 Soundkarte(n)	788
26.6.10 CLR-Version	789
26.7 ... den Code-Editor voll nutzen?	790
26.7.1 Refactoring/Umgestaltung	790
26.7.2 Surrounding	791
26.7.3 Automatisches Umbenennen von Formularen	791
26.7.4 Code-Schnipsel/Codeausschnitte	792

Teil V: Komplexbeispiele

27 Kleine Textverarbeitung	795
27.1 Oberflächenentwurf	796
27.1.1 Das MDI-Rahmenfenster	796
27.1.2 MDI-Kindfenster	797
27.1.3 Menüs zusammenführen	798
27.1.4 Ein PopUp-Menü hinzufügen	800
27.2 Quellcode für MDI-Hauptfenster	801
27.2.1 Datei-Menü	801
27.2.2 Fenster-Menü	802
27.2.3 Hilfe-Menü	802
27.3 Quellcode für MDI-Kindfenster	803
27.3.1 Datei-Menü	804
27.3.2 Bearbeiten-Menü	805
27.3.3 Zeichen-Menü	805
27.3.4 PopUp-Menü programmieren	806
27.3.5 Programmtest	806
27.4 Dokumente drucken	807
27.4.1 Ergänzungen der Oberfläche der Kindfenster	807
27.4.2 Ergänzungen zum Quellcode der ChildForm	808
27.4.3 Test	810
27.4.4 Bemerkungen zur RichTextBox	812
28 Wissenschaftlicher Rechner	813
28.1 Basisversion des Rechners	814
28.1.1 Entwurf der Benutzerschnittstelle	814
28.1.2 Klasse Calculator	815
28.1.3 Quellcode Form1	816
28.1.4 Test	817
28.2 Die Assembly als Datei speichern	819
28.2.1 Oberfläche	819
28.2.2 Quelltext (Form1)	819
28.2.3 Test	821
28.3 Berechnungsergebnisse als Diagramm darstellen	822
28.3.1 Oberfläche	822
28.3.2 Die Klasse CalculatorX	823

28.3.3	Quellcode Form1	824
28.3.4	Test	825
29	Datenverwaltung mittels Random Access Datei	827
29.1	Vorbereitungen	828
29.1.1	Klassenübersicht	828
29.1.2	Entwurf der Benutzerschnittstelle	828
29.2	Programmieren des Geschäftsmodells	829
29.2.1	Klasse CRandomKunde	829
29.2.2	Klasse CNewRandomFile	831
29.2.3	Klasse CRandomAccess	832
29.3	Abschließende Arbeiten	837
29.3.1	Programmieren der Bedienoberfläche	837
29.3.2	Programmtest	839
30	Verkehrsmanagement per Multithreading	841
30.1	Etwas Theorie	842
30.1.1	Kurzeinführung Threading	842
30.1.2	Wichtige Thread-Methoden	843
30.1.3	Wichtige Thread-Eigenschaften	844
30.2	Vorbereitungen	845
30.2.1	Aufgabenstellung	845
30.2.2	Klassenübersicht	845
30.2.3	Oberfläche	845
30.3	Programmierung	846
30.3.1	Klasse LKW	846
30.3.2	Klasse Schiff	849
30.3.3	Klasse Controller	852
30.3.4	Klasse Global	854
30.3.5	Klasse Form1	854
30.3.6	Test	857
Register	859