

walter DOBERENZ
thomas GEWINNUS



Visual C# 2015

GRUNDLAGEN
PROFIWISSEN
REZEPTE

// C#-Grundlagen
// LINQ, OOP, ADO.NET
// App-Entwicklung
// Über 150 Praxisbeispiele



EXTRA: 700 Seiten Bonuskapitel
zu WPF und Windows Forms

HANSER

Doberenz/Gewinnus

Visual C# 2015 Grundlagen, Profiwissen und Rezepte

Bleiben Sie auf dem Laufenden!



Unser **Computerbuch-Newsletter** informiert Sie monatlich über neue Bücher und Termine. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter



www.hanser-fachbuch.de/newsletter



Hanser Update ist der IT-Blog des Hanser Verlags mit Beiträgen und Praxistipps von unseren Autoren rund um die Themen Online Marketing, Webentwicklung, Programmierung, Softwareentwicklung sowie IT- und Projektmanagement. Lesen Sie mit und abonnieren Sie unsere News unter



www.hanser-fachbuch.de/update



Walter Doberenz
Thomas Gewinnus

Visual C# 2015

Grundlagen, Profiwissen
und Rezepte

HANSER

Die Autoren:

Professor Dr.-Ing. habil. Walter Doberenz, Wintersdorf

Dipl.-Ing. Thomas Gewinnus, Frankfurt/Oder

Alle in diesem Buch enthaltenen Informationen, Verfahren und Darstellungen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autoren und Verlag übernehmen infolgedessen keine juristische Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieser Informationen – oder Teilen davon – entsteht.

Ebenso übernehmen Autoren und Verlag keine Gewähr dafür, dass beschriebene Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Buch berechtigt deshalb auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.



Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdrucks und der Vervielfältigung des Buches, oder Teilen daraus, sind vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren), auch nicht für Zwecke der Unterrichtsgestaltung, reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

© 2015 Carl Hanser Verlag München

<http://www.hanser-fachbuch.de>

Lektorat: Sieglinde Schärl

Herstellung: Irene Weilhart

Satz: Ingenieurbüro Gewinnus

Sprachlektorat: Walter Doberenz

Umschlagdesign: Marc Müller-Bremer, www.rebranding.de, München

Umschlagrealisation: Stephan Rönigk

Druck und Bindung: Kösel, Krugzell

Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702

Printed in Germany

Print-ISBN: 978-3-446-44381-5

E-Book-ISBN: 978-3-446-44606-9

Inhaltsverzeichnis

Vorwort	45
 Teil I: Grundlagen	
1 Einstieg in Visual Studio 2015	51
1.1 Die Installation von Visual Studio 2015	51
1.1.1 Überblick über die Produktpalette	51
1.1.2 Anforderungen an Hard- und Software	52
1.2 Unser allererstes C#-Programm	53
1.2.1 Vorbereitungen	53
1.2.2 Quellcode schreiben	55
1.2.3 Programm kompilieren und testen	55
1.2.4 Einige Erläuterungen zum Quellcode	56
1.2.5 Konsolenanwendungen sind out	57
1.3 Die Windows-Philosophie	57
1.3.1 Mensch-Rechner-Dialog	57
1.3.2 Objekt- und ereignisorientierte Programmierung	58
1.3.3 Programmieren mit Visual Studio 2015	59
1.4 Die Entwicklungsumgebung Visual Studio 2015	60
1.4.1 Neues Projekt	61
1.4.2 Die wichtigsten Fenster	62
1.5 Microsofts .NET-Technologie	65
1.5.1 Zur Geschichte von .NET	65
1.5.2 .NET-Features und Begriffe	67
1.6 Wichtige Neuigkeiten in Visual Studio 2015	74
1.6.1 Entwicklungsumgebung	74
1.6.2 Neue C#-Sprachfeatures	74
1.6.3 Code-Editor	75
1.6.4 NET Framework 4.6	75

1.7	Praxisbeispiele	76
1.7.1	Unsere erste Windows Forms-Anwendung	76
1.7.2	Umrechnung Euro-Dollar	80
2	Grundlagen der Sprache C#	89
2.1	Grundbegriffe	89
2.1.1	Anweisungen	89
2.1.2	Bezeichner	90
2.1.3	Schlüsselwörter	91
2.1.4	Kommentare	91
2.2	Datentypen, Variablen und Konstanten	92
2.2.1	Fundamentale Typen	92
2.2.2	Werttypen versus Verweistypen	93
2.2.3	Benennung von Variablen	94
2.2.4	Deklaration von Variablen	94
2.2.5	Typsuffixe	96
2.2.6	Zeichen und Zeichenketten	97
2.2.7	object-Datentyp	99
2.2.8	Konstanten deklarieren	99
2.2.9	Nullable Types	100
2.2.10	Typinferenz	101
2.2.11	Gültigkeitsbereiche und Sichtbarkeit	102
2.3	Konvertieren von Datentypen	102
2.3.1	Implizite und explizite Konvertierung	102
2.3.2	Welcher Datentyp passt zu welchem?	104
2.3.3	Konvertieren von string	105
2.3.4	Die Convert-Klasse	107
2.3.5	Die Parse-Methode	107
2.3.6	Boxing und Unboxing	108
2.4	Operatoren	109
2.4.1	Arithmetische Operatoren	110
2.4.2	Zuweisungsoperatoren	111
2.4.3	Logische Operatoren	112
2.4.4	Rangfolge der Operatoren	115
2.5	Kontrollstrukturen	116
2.5.1	Verzweigungsbefehle	116
2.5.2	Schleifenanweisungen	119

2.6	Benutzerdefinierte Datentypen	122
2.6.1	Enumerationen	122
2.6.2	Strukturen	123
2.7	Nutzerdefinierte Methoden	125
2.7.1	Methoden mit Rückgabewert	126
2.7.2	Methoden ohne Rückgabewert	127
2.7.3	Parameterübergabe mit ref	128
2.7.4	Parameterübergabe mit out	129
2.7.5	Methodenüberladung	130
2.7.6	Optionale Parameter	131
2.7.7	Benannte Parameter	132
2.8	Praxisbeispiele	133
2.8.1	Vom PAP zur Konsolenanwendung	133
2.8.2	Ein Konsolen- in ein Windows-Programm verwandeln	135
2.8.3	Schleifenanweisungen verstehen	137
2.8.4	Benutzerdefinierte Methoden überladen	139
2.8.5	Anwendungen von Visual Basic nach C# portieren	142
3	OOP-Konzepte	149
3.1	Kleine Einführung in die OOP	149
3.1.1	Historische Entwicklung	149
3.1.2	Grundbegriffe der OOP	151
3.1.3	Sichtbarkeit von Klassen und ihren Mitgliedern	153
3.1.4	Allgemeiner Aufbau einer Klasse	154
3.1.5	Das Erzeugen eines Objekts	155
3.1.6	Einführungsbeispiel	158
3.2	Eigenschaften	163
3.2.1	Eigenschaften mit Zugriffsmethoden kapseln	163
3.2.2	Berechnete Eigenschaften	165
3.2.3	Lese-/Schreibschutz	167
3.2.4	Property-Accessoren	168
3.2.5	Statische Felder/Eigenschaften	168
3.2.6	Einfache Eigenschaften automatisch implementieren	171
3.3	Methoden	172
3.3.1	Öffentliche und private Methoden	172
3.3.2	Überladene Methoden	173
3.3.3	Statische Methoden	174

3.4	Ereignisse	176
3.4.1	Ereignis hinzufügen	176
3.4.2	Ereignis verwenden	179
3.5	Arbeiten mit Konstruktor und Destruktor	182
3.5.1	Konstruktor und Objektinitialisierer	182
3.5.2	Destruktor und Garbage Collector	185
3.5.3	Mit using den Lebenszyklus des Objekts kapseln	188
3.5.4	Verzögerte Initialisierung	188
3.6	Vererbung und Polymorphie	189
3.6.1	Klassendiagramm	189
3.6.2	Method-Overriding	190
3.6.3	Klassen implementieren	191
3.6.4	Implementieren der Objekte	194
3.6.5	Ausblenden von Mitgliedern durch Vererbung	196
3.6.6	Allgemeine Hinweise und Regeln zur Vererbung	197
3.6.7	Polymorphes Verhalten	199
3.6.8	Die Rolle von System.Object	201
3.7	Spezielle Klassen	202
3.7.1	Abstrakte Klassen	202
3.7.2	Versiegelte Klassen	204
3.7.3	Partielle Klassen	204
3.7.4	Statische Klassen	205
3.8	Schnittstellen (Interfaces)	206
3.8.1	Definition einer Schnittstelle	207
3.8.2	Implementieren einer Schnittstelle	207
3.8.3	Abfragen, ob Schnittstelle vorhanden ist	208
3.8.4	Mehrere Schnittstellen implementieren	209
3.8.5	Schnittstellenprogrammierung ist ein weites Feld	209
3.9	Praxisbeispiele	209
3.9.1	Eigenschaften sinnvoll kapseln	209
3.9.2	Eine statische Klasse anwenden	212
3.9.3	Vom fetten zum schlanken Client	214
3.9.4	Schnittstellenvererbung verstehen	226
3.9.5	Rechner für komplexe Zahlen	231
3.9.6	Formel-Rechner mit dem CodeDOM	240
3.9.7	Berechnungsergebnisse als Diagramm darstellen	248
3.9.8	Sortieren mit IComparable/IComparer	252
3.9.9	Einen Objektbaum in generischen Listen abspeichern	257

3.9.10	OOO beim Kartenspiel erlernen	263
3.9.11	Eine Klasse zur Matrizenrechnung entwickeln	267
4	Arrays, Strings, Funktionen	273
4.1	Datenfelder (Arrays)	273
4.1.1	Array deklarieren	273
4.1.2	Array instanzieren	274
4.1.3	Array initialisieren	274
4.1.4	Zugriff auf Array-Elemente	275
4.1.5	Zugriff mittels Schleife	276
4.1.6	Mehrdimensionale Arrays	277
4.1.7	Zuweisen von Arrays	279
4.1.8	Arrays aus Strukturvariablen	280
4.1.9	Löschen und Umdimensionieren von Arrays	281
4.1.10	Eigenschaften und Methoden von Arrays	282
4.1.11	Übergabe von Arrays	284
4.2	Verarbeiten von Zeichenketten	285
4.2.1	Zuweisen von Strings	285
4.2.2	Eigenschaften und Methoden von String-Variablen	286
4.2.3	Wichtige Methoden der String-Klasse	288
4.2.4	Die StringBuilder-Klasse	290
4.3	Reguläre Ausdrücke	292
4.3.1	Wozu werden reguläre Ausdrücke verwendet?	293
4.3.2	Eine kleine Einführung	293
4.3.3	Wichtige Methoden/Eigenschaften der Klasse Regex	294
4.3.4	Kompilierte reguläre Ausdrücke	296
4.3.5	RegexOptions-Enumeration	297
4.3.6	Metazeichen (Escape-Zeichen)	297
4.3.7	Zeichenmengen (Character Sets)	298
4.3.8	Quantifizierer	300
4.3.9	Zero-Width Assertions	301
4.3.10	Gruppen	304
4.3.11	Text ersetzen	305
4.3.12	Text splitten	306
4.4	Datums- und Zeitberechnungen	307
4.4.1	Die DateTime-Struktur	307
4.4.2	Wichtige Eigenschaften von DateTime-Variablen	308
4.4.3	Wichtige Methoden von DateTime-Variablen	309

4.4.4	Wichtige Mitglieder der DateTime-Struktur	309
4.4.5	Konvertieren von Datumstrings in DateTime-Werte	310
4.4.6	Die TimeSpan-Struktur	311
4.5	Mathematische Funktionen	312
4.5.1	Überblick	312
4.5.2	Zahlen runden	313
4.5.3	Winkel umrechnen	313
4.5.4	Potenz- und Wurzeloperationen	313
4.5.5	Logarithmus und Exponentialfunktionen	314
4.5.6	Zufallszahlen erzeugen	314
4.6	Zahlen- und Datumsformatierungen	315
4.6.1	Anwenden der ToString-Methode	315
4.6.2	Anwenden der Format-Methode	317
4.6.3	Stringinterpolation	318
4.7	Praxisbeispiele	319
4.7.1	Zeichenketten verarbeiten	319
4.7.2	Zeichenketten mit StringBuilder addieren	322
4.7.3	Reguläre Ausdrücke testen	325
4.7.4	Methodenaufrufe mit Array-Parametern	327
5	Weitere Sprachfeatures	331
5.1	Namespaces (Namensräume)	331
5.1.1	Ein kleiner Überblick	331
5.1.2	Einen eigenen Namespace einrichten	332
5.1.3	Die using-Anweisung	333
5.1.4	Namespace Alias	334
5.1.5	Namespace Alias Qualifizierer	334
5.2	Operatorenüberladung	335
5.2.1	Syntaxregeln	335
5.2.2	Praktische Anwendung	336
5.3	Collections (Auflistungen)	337
5.3.1	Die Schnittstelle IEnumerable	337
5.3.2	ArrayList	339
5.3.3	Hashtable	341
5.3.4	Indexer	341
5.4	Generics	343
5.4.1	Klassische Vorgehensweise	344
5.4.2	Generics bieten Typsicherheit	345

5.4.3	Generische Methoden	346
5.4.4	Iteratoren	347
5.5	Generische Collections	348
5.5.1	List-Collection statt ArrayList	348
5.5.2	Vorteile generischer Collections	349
5.5.3	Constraints	349
5.6	Das Prinzip der Delegates	349
5.6.1	Delegates sind Methodenzeiger	350
5.6.2	Einen Delegate-Typ deklarieren	350
5.6.3	Ein Delegate-Objekt erzeugen	350
5.6.4	Delegates vereinfacht instanziiieren	352
5.6.5	Anonyme Methoden	353
5.6.6	Lambda-Ausdrücke	354
5.6.7	Lambda-Ausdrücke in der Task Parallel Library	356
5.7	Dynamische Programmierung	358
5.7.1	Wozu dynamische Programmierung?	358
5.7.2	Das Prinzip der dynamischen Programmierung	358
5.7.3	Optionale Parameter sind hilfreich	361
5.7.4	Kovarianz und Kontravarianz	362
5.8	Weitere Datentypen	362
5.8.1	BigInteger	362
5.8.2	Complex	365
5.8.3	Tuple<>	365
5.8.4	SortedSet<>	366
5.9	Praxisbeispiele	367
5.9.1	ArrayList versus generische List	367
5.9.2	Generische IEnumerable-Interfaces implementieren	371
5.9.3	Delegates, anonyme Methoden, Lambda Expressions	374
5.9.4	Dynamischer Zugriff auf COM Interop	378
6	Einführung in LINQ	381
6.1	LINQ-Grundlagen	381
6.1.1	Die LINQ-Architektur	381
6.1.2	Anonyme Typen	382
6.1.3	Erweiterungsmethoden	384
6.2	Abfragen mit LINQ to Objects	385
6.2.1	Grundlegendes zur LINQ-Syntax	385
6.2.2	Zwei alternative Schreibweisen von LINQ Abfragen	386

6.2.3	Übersicht der wichtigsten Abfrage-Operatoren	387
6.3	LINQ-Abfragen im Detail	388
6.3.1	Die Projektionsoperatoren Select und SelectMany	389
6.3.2	Der Restriktionsoperator Where	390
6.3.3	Die Sortierungsoperatoren OrderBy und ThenBy	391
6.3.4	Der Gruppierungsoperator GroupBy	392
6.3.5	Verknüpfen mit Join	395
6.3.6	Aggregat-Operatoren	395
6.3.7	Verzögertes Ausführen von LINQ-Abfragen	397
6.3.8	Konvertierungsmethoden	398
6.3.9	Abfragen mit PLINQ	398
6.4	Praxisbeispiele	401
6.4.1	Die Syntax von LINQ-Abfragen verstehen	401
6.4.2	Aggregat-Abfragen mit LINQ	404
6.4.3	LINQ im Schnelldurchgang erlernen	407
6.4.4	Strings mit LINQ abfragen und filtern	409
6.4.5	Duplikate aus einer Liste oder einem Array entfernen	410
6.4.6	Arrays mit LINQ initialisieren	413
6.4.7	Arrays per LINQ mit Zufallszahlen füllen	415
6.4.8	Einen String mit Wiederholmuster erzeugen	417
6.4.9	Mit LINQ Zahlen und Strings sortieren	418
6.4.10	Mit LINQ Collections von Objekten sortieren	419
6.4.11	Ergebnisse von LINQ-Abfragen in ein Array kopieren	422

Teil II: Technologien

7	Zugriff auf das Dateisystem	425
7.1	Grundlagen	425
7.1.1	Klassen für den Zugriff auf das Dateisystem	426
7.1.2	Statische versus Instanzen-Klasse	426
7.2	Übersichten	427
7.2.1	Methoden der Directory-Klasse	427
7.2.2	Methoden eines DirectoryInfo-Objekts	428
7.2.3	Eigenschaften eines DirectoryInfo-Objekts	428
7.2.4	Methoden der File-Klasse	428
7.2.5	Methoden eines FileInfo-Objekts	429
7.2.6	Eigenschaften eines FileInfo-Objekts	430

7.3	Operationen auf Verzeichnisebene	430
7.3.1	Existenz eines Verzeichnisses/einer Datei feststellen	430
7.3.2	Verzeichnisse erzeugen und löschen	431
7.3.3	Verzeichnisse verschieben und umbenennen	431
7.3.4	Aktuelles Verzeichnis bestimmen	432
7.3.5	Unterverzeichnisse ermitteln	432
7.3.6	Alle Laufwerke ermitteln	432
7.3.7	Dateien kopieren und verschieben	433
7.3.8	Dateien umbenennen	434
7.3.9	Dateiattribute feststellen	434
7.3.10	Verzeichnis einer Datei ermitteln	436
7.3.11	Alle im Verzeichnis enthaltenen Dateien ermitteln	436
7.3.12	Dateien und Unterverzeichnisse ermitteln	436
7.4	Zugriffsberechtigungen	437
7.4.1	ACL und ACE	437
7.4.2	SetAccessControl-Methode	438
7.4.3	Zugriffsrechte anzeigen	438
7.5	Weitere wichtige Klassen	439
7.5.1	Die Path-Klasse	439
7.5.2	Die Klasse FileSystemWatcher	440
7.6	Datei- und Verzeichnisdialoge	441
7.6.1	OpenFileDialog und SaveFileDialog	442
7.6.2	FolderBrowserDialog	443
7.7	Praxisbeispiele	444
7.7.1	Infos über Verzeichnisse und Dateien gewinnen	444
7.7.2	Eine Verzeichnisstruktur in die TreeView einlesen	448
7.7.3	Mit LINQ und RegEx Verzeichnisbäume durchsuchen	450
8	Dateien lesen und schreiben	455
8.1	Grundprinzip der Datenpersistenz	455
8.1.1	Dateien und Streams	455
8.1.2	Die wichtigsten Klassen	456
8.1.3	Erzeugen eines Streams	457
8.2	Dateiparameter	457
8.2.1	FileAccess	457
8.2.2	FileMode	457
8.2.3	FileShare	458

8.3	Textdateien	458
8.3.1	Eine Textdatei beschreiben bzw. neu anlegen	458
8.3.2	Eine Textdatei lesen	460
8.4	Binärdateien	461
8.4.1	Lese-/Schreibzugriff	461
8.4.2	Die Methoden ReadAllBytes und WriteAllBytes	462
8.4.3	Erzeugen von BinaryReader/BinaryWriter	462
8.5	Sequenzielle Dateien	463
8.5.1	Lesen und schreiben von strukturierten Daten	463
8.5.2	Serialisieren von Objekten	464
8.6	Dateien verschlüsseln und komprimieren	465
8.6.1	Das Methodenpärchen Encrypt/Decrypt	465
8.6.2	Verschlüsseln unter Windows Vista/7/8/10	465
8.6.3	Verschlüsseln mit der CryptoStream-Klasse	466
8.6.4	Dateien komprimieren	467
8.7	Memory Mapped Files	468
8.7.1	Grundprinzip	468
8.7.2	Erzeugen eines MMF	469
8.7.3	Erstellen eines Map View	470
8.8	Praxisbeispiele	471
8.8.1	Auf eine Textdatei zugreifen	471
8.8.2	Einen Objektbaum persistent speichern	474
8.8.3	Ein Memory Mapped File (MMF) verwenden	481
8.8.4	Hex-Dezimal-Bytes-Konverter	483
8.8.5	Eine Datei verschlüsseln	487
8.8.6	Eine Datei komprimieren	490
8.8.7	Echte ZIP-Dateien erstellen	492
8.8.8	PDFs erstellen/exportieren	494
8.8.9	Eine CSV-Datei erstellen	497
8.8.10	Eine CSV-Datei mit LINQ lesen und auswerten	500
8.8.11	Einen korrekten Dateinamen erzeugen	503
9	Asynchrone Programmierung	505
9.1	Übersicht	505
9.1.1	Multitasking versus Multithreading	506
9.1.2	Deadlocks	507
9.1.3	Racing	507

9.2	Programmieren mit Threads	509
9.2.1	Einführungsbeispiel	509
9.2.2	Wichtige Thread-Methoden	510
9.2.3	Wichtige Thread-Eigenschaften	512
9.2.4	Einsatz der ThreadPool-Klasse	513
9.3	Sperrmechanismen	515
9.3.1	Threading ohne lock	515
9.3.2	Threading mit lock	517
9.3.3	Die Monitor-Klasse	519
9.3.4	Mutex	522
9.3.5	Methoden für die parallele Ausführung sperren	524
9.3.6	Semaphore	524
9.4	Interaktion mit der Programmoberfläche	526
9.4.1	Die Werkzeuge	526
9.4.2	Einzelne Steuerelemente mit Invoke aktualisieren	526
9.4.3	Mehrere Steuerelemente aktualisieren	528
9.4.4	Ist ein Invoke-Aufruf nötig?	528
9.4.5	Und was ist mit WPF?	529
9.5	Timer-Threads	530
9.6	Die BackgroundWorker-Komponente	531
9.7	Asynchrone Programmier-Entwurfsmuster	534
9.7.1	Kurzübersicht	534
9.7.2	Polling	535
9.7.3	Callback verwenden	537
9.7.4	Callback mit Parameterübergabe verwenden	538
9.7.5	Callback mit Zugriff auf die Programm-Oberfläche	539
9.8	Asynchroner Aufruf beliebiger Methoden	540
9.8.1	Die Beispielklasse	540
9.8.2	Asynchroner Aufruf ohne Callback	542
9.8.3	Asynchroner Aufruf mit Callback und Anzeigefunktion	543
9.8.4	Aufruf mit Rückgabewerten (per Eigenschaft)	544
9.8.5	Aufruf mit Rückgabewerten (per EndInvoke)	544
9.9	Es geht auch einfacher – async und await	545
9.9.1	Der Weg von Synchron zu Asynchron	546
9.9.2	Achtung: Fehlerquellen!	548
9.9.3	Eigene asynchrone Methoden entwickeln	550

9.10	Praxisbeispiele	552
9.10.1	Spieltrieb & Multithreading erleben	552
9.10.2	Prozess- und Thread-Informationen gewinnen	565
9.10.3	Ein externes Programm starten	570
10	Die Task Parallel Library	573
10.1	Überblick	573
10.1.1	Parallel-Programmierung	573
10.1.2	Möglichkeiten der TPL	576
10.1.3	Der CLR-Threadpool	576
10.2	Parallele Verarbeitung mit Parallel.Invoke	577
10.2.1	Aufrufvarianten	578
10.2.2	Einschränkungen	579
10.3	Verwendung von Parallel.For	579
10.3.1	Abbrechen der Verarbeitung	581
10.3.2	Auswerten des Bearbeitungsstatus	582
10.3.3	Und was ist mit anderen Iterator-Schrittweiten?	583
10.4	Collections mit Parallel.ForEach verarbeiten	583
10.5	Die Task-Klasse	584
10.5.1	Einen Task erzeugen	584
10.5.2	Den Task starten	585
10.5.3	Datenübergabe an den Task	587
10.5.4	Wie warte ich auf das Ende des Task?	588
10.5.5	Tasks mit Rückgabewerten	590
10.5.6	Die Verarbeitung abbrechen	593
10.5.7	Fehlerbehandlung	596
10.5.8	Weitere Eigenschaften	597
10.6	Zugriff auf das User-Interface	598
10.6.1	Task-Ende und Zugriff auf die Oberfläche	599
10.6.2	Zugriff auf das UI aus dem Task heraus	600
10.7	Weitere Datenstrukturen im Überblick	602
10.7.1	Thread sichere Collections	602
10.7.2	Primitive für die Threadsynchronisation	603
10.8	Parallel LINQ (PLINQ)	603
10.9	Praxisbeispiel: Spieltrieb – Version 2	604
10.9.1	Aufgabenstellung	604
10.9.2	Global-Klasse	604
10.9.3	Controller-Klasse	606

10.9.4	LKW-Klasse	607
10.9.5	Schiff-Klasse	609
10.9.6	Oberfläche	611
11	Fehlersuche und Behandlung	613
11.1	Der Debugger	613
11.1.1	Allgemeine Beschreibung	613
11.1.2	Die wichtigsten Fenster	614
11.1.3	Debugging-Optionen	617
11.1.4	Praktisches Debugging am Beispiel	619
11.2	Arbeiten mit Debug und Trace	623
11.2.1	Wichtige Methoden von Debug und Trace	623
11.2.2	Besonderheiten der Trace-Klasse	627
11.2.3	TraceListener-Objekte	627
11.3	Caller Information	630
11.3.1	Attribute	630
11.3.2	Anwendung	630
11.4	Fehlerbehandlung	631
11.4.1	Anweisungen zur Fehlerbehandlung	631
11.4.2	try-catch	632
11.4.3	try-finally	637
11.4.4	Das Standardverhalten bei Ausnahmen festlegen	639
11.4.5	Die Exception-Klasse	640
11.4.6	Fehler/Ausnahmen auslösen	641
11.4.7	Eigene Fehlerklassen	641
11.4.8	Exceptionhandling zur Entwurfszeit	643
11.4.9	Code Contracts	644
12	XML in Theorie und Praxis	645
12.1	XML – etwas Theorie	645
12.1.1	Übersicht	645
12.1.2	Der XML-Grundaufbau	648
12.1.3	Wohlgeformte Dokumente	649
12.1.4	Processing Instructions (PI)	652
12.1.5	Elemente und Attribute	652
12.1.6	Verwendbare Zeichensätze	654

12.2	XSD-Schemas	656
12.2.1	XSD-Schemas und ADO.NET	656
12.2.2	XML-Schemas in Visual Studio analysieren	658
12.2.3	XML-Datei mit XSD-Schema erzeugen	661
12.2.4	XSD-Schema aus einer XML-Datei erzeugen	662
12.3	Verwendung des DOM unter .NET	663
12.3.1	Übersicht	663
12.3.2	DOM-Integration in C#	664
12.3.3	Laden von Dokumenten	664
12.3.4	Erzeugen von XML-Dokumenten	665
12.3.5	Auslesen von XML-Dateien	667
12.3.6	Direktzugriff auf einzelne Elemente	669
12.3.7	Einfügen von Informationen	669
12.3.8	Suchen in den Baumzweigen	672
12.4	XML-Verarbeitung mit LINQ to XML	675
12.4.1	Die LINQ to XML-API	675
12.4.2	Neue XML-Dokumente erzeugen	677
12.4.3	Laden und Sichern von XML-Dokumenten	679
12.4.4	Navigieren in XML-Daten	680
12.4.5	Auswählen und Filtern	682
12.4.6	Manipulieren der XML-Daten	682
12.4.7	XML-Dokumente transformieren	684
12.5	Weitere Möglichkeiten der XML-Verarbeitung	687
12.5.1	Die relationale Sicht mit XmlDataDocument	687
12.5.2	XML-Daten aus Objektstrukturen erzeugen	690
12.5.3	Schnelles Suchen in XML-Daten mit XPathNavigator	693
12.5.4	Schnelles Auslesen von XML-Daten mit XmlReader	696
12.5.5	Erzeugen von XML-Daten mit XmlWriter	698
12.5.6	XML transformieren mit XSLT	700
12.6	Praxisbeispiele	702
12.6.1	Mit dem DOM in XML-Dokumenten navigieren	702
12.6.2	XML-Daten in eine TreeView einlesen	705
12.6.3	Ein DataSet in einen XML-String konvertieren	709
12.6.4	Ein DataSet in einer XML-Datei speichern	713
12.6.5	In Dokumenten mit dem XPathNavigator navigieren	716

13	Einführung in ADO.NET	721
13.1	Eine kleine Übersicht	721
13.1.1	Die ADO.NET-Klassenhierarchie	721
13.1.2	Die Klassen der Datenprovider	722
13.1.3	Das Zusammenspiel der ADO.NET-Klassen	725
13.2	Das Connection-Objekt	726
13.2.1	Allgemeiner Aufbau	726
13.2.2	OleDbConnection	726
13.2.3	Schließen einer Verbindung	728
13.2.4	Eigenschaften des Connection-Objekts	728
13.2.5	Methoden des Connection-Objekts	730
13.2.6	Der SqlConnectionStringBuilder	731
13.3	Das Command-Objekt	731
13.3.1	Erzeugen und Anwenden eines Command-Objekts	732
13.3.2	Erzeugen mittels CreateCommand-Methode	732
13.3.3	Eigenschaften des Command-Objekts	733
13.3.4	Methoden des Command-Objekts	735
13.3.5	Freigabe von Connection- und Command-Objekten	736
13.4	Parameter-Objekte	737
13.4.1	Erzeugen und Anwenden eines Parameter-Objekts	737
13.4.2	Eigenschaften des Parameter-Objekts	738
13.5	Das SqlCommandBuilder-Objekt	739
13.5.1	Erzeugen	739
13.5.2	Anwenden	739
13.6	Das SqlDataReader-Objekt	740
13.6.1	SqlDataReader erzeugen	740
13.6.2	Daten lesen	741
13.6.3	Eigenschaften des SqlDataReader	742
13.6.4	Methoden des SqlDataReader	742
13.7	Das SqlDataAdapter-Objekt	743
13.7.1	DataAdapter erzeugen	743
13.7.2	Command-Eigenschaften	744
13.7.3	Fill-Methode	745
13.7.4	Update-Methode	746
13.8	Praxisbeispiele	747
13.8.1	Wichtige ADO.NET-Objekte im Einsatz	747
13.8.2	Eine Aktionsabfrage ausführen	748

13.8.3	Eine Auswahlabfrage aufrufen	751
13.8.4	Die Datenbank aktualisieren	753
14	Das DataSet	757
14.1	Grundlegende Features des DataSets	757
14.1.1	Die Objekthierarchie	758
14.1.2	Die wichtigsten Klassen	758
14.1.3	Erzeugen eines DataSets	759
14.2	Das DataTable-Objekt	760
14.2.1	DataTable erzeugen	761
14.2.2	Spalten hinzufügen	761
14.2.3	Zeilen zur DataTable hinzufügen	762
14.2.4	Auf den Inhalt einer DataTable zugreifen	763
14.3	Die DataView	765
14.3.1	Erzeugen einer DataView	765
14.3.2	Sortieren und Filtern von Datensätzen	765
14.3.3	Suchen von Datensätzen	766
14.4	Typisierte DataSets	766
14.4.1	Ein typisiertes DataSet erzeugen	767
14.4.2	Das Konzept der Datenquellen	768
14.4.3	Typisierte DataSets und TableAdapter	769
14.5	Die Qual der Wahl	770
14.5.1	DataReader – der schnelle Lesezugriff	771
14.5.2	DataSet – die Datenbank im Hauptspeicher	771
14.5.3	Objektrelationales Mapping – die Zukunft?	772
14.6	Praxisbeispiele	773
14.6.1	In der DataView sortieren und filtern	773
14.6.2	Suche nach Datensätzen	775
14.6.3	Ein DataSet in einen XML-String serialisieren	776
14.6.4	Untypisiertes in ein typisiertes DataSet konvertieren	781
14.6.5	Eine LINQ to SQL-Abfrage ausführen	786
15	Verteilen von Anwendungen	791
15.1	ClickOnce-Deployment	792
15.1.1	Übersicht/Einschränkungen	792
15.1.2	Die Vorgehensweise	793
15.1.3	Ort der Veröffentlichung	793
15.1.4	Anwendungsdateien	794

15.1.5	Erforderliche Komponenten	794
15.1.6	Aktualisierungen	795
15.1.7	Veröffentlichungsoptionen	796
15.1.8	Veröffentlichen	797
15.1.9	Verzeichnisstruktur	797
15.1.10	Der Webpublishing-Assistent	799
15.1.11	Neue Versionen erstellen	800
15.2	InstallShield	800
15.2.1	Installation	800
15.2.2	Aktivieren	801
15.2.3	Ein neues Setup-Projekt	801
15.2.4	Finaler Test	809
15.3	Hilfdateien programmieren	809
15.3.1	Der HTML Help Workshop	810
15.3.2	Bedienung am Beispiel	811
15.3.3	Hilfdateien in die Visual C#-Anwendung einbinden	813
15.3.4	Eine alternative Hilfe-IDE verwenden	817
16	Weitere Techniken	819
16.1	Zugriff auf die Zwischenablage	819
16.1.1	Das Clipboard-Objekt	819
16.1.2	Zwischenablage-Funktionen für Textboxen	821
16.2	Arbeiten mit der Registry	821
16.2.1	Allgemeines	822
16.2.2	Registry-Unterstützung in .NET	824
16.3	.NET-Reflection	825
16.3.1	Übersicht	825
16.3.2	Assembly laden	825
16.3.3	Mittels GetType und Type Informationen sammeln	826
16.3.4	Dynamisches Laden von Assemblies	828
16.4	Praxisbeispiele	831
16.4.1	Zugriff auf die Registry	831
16.4.2	Dateiverknüpfungen erzeugen	833
16.4.3	Betrachter für Manifestressourcen	835
16.4.4	Die Zwischenablage überwachen und anzeigen	838
16.4.5	Die WIA-Library kennenlernen	841
16.4.6	Auf eine Webcam zugreifen	853
16.4.7	Auf den Scanner zugreifen	855

16.4.8	OpenOffice.org Writer per OLE steuern	860
16.4.9	Nutzer und Gruppen des aktuellen Systems ermitteln	868
16.4.10	Testen, ob Nutzer in einer Gruppe enthalten ist	869
16.4.11	Testen, ob der Nutzer ein Administrator ist	871
16.4.12	Die IP-Adressen des Computers bestimmen	872
16.4.13	Die IP-Adresse über den Hostnamen bestimmen	873
16.4.14	Diverse Systeminformationen ermitteln	874
16.4.15	Environment Variablen auslesen	878
16.4.16	Alles über den Bildschirm erfahren	882
16.4.17	Sound per MCI aufnehmen	883
16.4.18	Mikrofonpegel anzeigen	887
16.4.19	Pegeldiagramm aufzeichnen	888
16.4.20	Sound-und Video-Dateien per MCI abspielen	892
17	Konsolenanwendungen	901
17.1	Grundaufbau/Konzepte	901
17.1.1	Unser Hauptprogramm – Program.cs	902
17.1.2	Rückgabe eines Fehlerstatus	903
17.1.3	Parameterübergabe	904
17.1.4	Zugriff auf die Umgebungsvariablen	905
17.2	Die Kommandozentrale: System.Console	906
17.2.1	Eigenschaften	907
17.2.2	Methoden/Ereignisse	907
17.2.3	Textausgaben	908
17.2.4	Farbangaben	909
17.2.5	Tastaturabfragen	910
17.2.6	Arbeiten mit Streamdaten	911
17.3	Praxisbeispiel	913
17.3.1	Farbige Konsolenanwendung	913
17.3.2	Weitere Hinweise und Beispiele	915

Teil III: Windows Apps

18	Erste Schritte	919
18.1	Grundkonzepte und Begriffe	919
18.1.1	Windows Runtime (WinRT)	919
18.1.2	Windows Apps	920

18.1.3	Fast and Fluid	921
18.1.4	Process Sandboxing und Contracts	922
18.1.5	.NET WinRT-Profil	924
18.1.6	Language Projection	924
18.1.7	Vollbildmodus – War da was?	926
18.1.8	Windows Store	926
18.1.9	Zielplattformen	927
18.2	Entwurfsumgebung	928
18.2.1	Betriebssystem	928
18.2.2	Windows-Simulator	928
18.2.3	Remote-Debugging	931
18.3	Ein (kleines) Einstiegsbeispiel	932
18.3.1	Aufgabenstellung	932
18.3.2	Quellcode	932
18.3.3	Oberflächenentwurf	935
18.3.4	Installation und Test	937
18.3.5	Touchscreen	939
18.3.6	Fazit	939
18.4	Weitere Details zu WinRT	941
18.4.1	Wo ist WinRT einzuordnen?	942
18.4.2	Die WinRT-API	943
18.4.3	Wichtige WinRT-Namespaces	945
18.4.4	Der Unterbau	946
18.5	Praxisbeispiel	948
18.5.1	WinRT in Desktop-Applikationen nutzen	948
19	App-Oberflächen entwerfen	953
19.1	Grundkonzepte	953
19.1.1	XAML (oder HTML 5) für die Oberfläche	954
19.1.2	Die Page, der Frame und das Window	955
19.1.3	Das Befehlsdesign	957
19.1.4	Die Navigationsdesigns	959
19.1.5	Achtung: Fingereingabe!	960
19.1.6	Verwendung von Schriftarten	960
19.2	Seitenauswahl und -navigation	961
19.2.1	Die Startseite festlegen	961
19.2.2	Navigation und Parameterübergabe	961
19.2.3	Den Seitenstatus erhalten	962

19.3	App-Darstellung	963
19.3.1	Vollbild quer und hochkant	963
19.3.2	Was ist mit Andocken und Füllmodus?	964
19.3.3	Reagieren auf die Änderung	964
19.4	Skalieren von Apps	966
19.5	Praxisbeispiele	969
19.5.1	Seitennavigation und Parameterübergabe	969
19.5.2	Die Fensterkopfzeile anpassen	971
20	Die wichtigsten Controls	973
20.1	Einfache WinRT-Controls	973
20.1.1	TextBlock, RichTextBlock	973
20.1.2	Button, HyperlinkButton, RepeatButton	976
20.1.3	CheckBox, RadioButton, ToggleButton, ToggleSwitch	978
20.1.4	TextBox, PasswordBox, RichEditBox	979
20.1.5	Image	983
20.1.6	ScrollBar, Slider, ProgressBar, ProgressRing	985
20.1.7	Border, Ellipse, Rectangle	986
20.2	Layout-Controls	987
20.2.1	Canvas	987
20.2.2	StackPanel	988
20.2.3	ScrollViewer	988
20.2.4	Grid	989
20.2.5	VariableSizedWrapGrid	990
20.2.6	SplitView	991
20.2.7	Pivot	995
20.2.8	RelativPanel	997
20.3	Listendarstellungen	999
20.3.1	ComboBox, ListBox	999
20.3.2	ListView	1003
20.3.3	GridView	1004
20.3.4	FlipView	1006
20.4	Sonstige Controls	1008
20.4.1	CaptureElement	1008
20.4.2	MediaElement	1010
20.4.3	Frame	1011
20.4.4	WebView	1011
20.4.5	ToolTip	1012

20.4.6	CalendarDatePicker	1014
20.4.7	DatePicker/TimePicker	1015
20.5	Praxisbeispiele	1015
20.5.1	Einen StringFormat-Konverter implementieren	1015
20.5.2	Besonderheiten der TextBox kennen lernen	1017
20.5.3	Daten in der GridView gruppieren	1021
20.5.4	Das SemanticZoom-Control verwenden	1025
20.5.5	Die CollectionViewSource verwenden	1030
20.5.6	Zusammenspiel ListBox/AppBar	1033
21	Apps im Detail	1037
21.1	Ein Windows App-Projekt im Detail	1037
21.1.1	Contracts und Extensions	1038
21.1.2	AssemblyInfo.cs	1038
21.1.3	Verweise	1040
21.1.4	App.xaml und App.xaml.cs	1040
21.1.5	Package.appxmanifest	1041
21.1.6	Application1_TemporaryKey.pfx	1046
21.1.7	MainPage.xaml & MainPage.xaml.cs	1046
21.1.8	Assets/Symbole	1047
21.1.9	Nach dem Kompilieren	1047
21.2	Der Lebenszyklus einer Windows App	1047
21.2.1	Möglichkeiten der Aktivierung von Apps	1049
21.2.2	Der Splash Screen	1051
21.2.3	Suspending	1051
21.2.4	Resuming	1053
21.2.5	Beenden von Apps	1053
21.2.6	Die Ausnahmen von der Regel	1054
21.2.7	Debuggen	1054
21.3	Daten speichern und laden	1058
21.3.1	Grundsätzliche Überlegungen	1058
21.3.2	Worauf und wie kann ich zugreifen?	1058
21.3.3	Das AppData-Verzeichnis	1059
21.3.4	Das Anwendungs-Installationsverzeichnis	1061
21.3.5	Das Downloads-Verzeichnis	1062
21.3.6	Sonstige Verzeichnisse	1063
21.3.7	Anwendungsdaten lokal sichern und laden	1063
21.3.8	Daten in der Cloud ablegen/laden (Roaming)	1065

21.3.9	Aufräumen	1067
21.3.10	Sensible Informationen speichern	1068
21.4	Praxisbeispiele	1069
21.4.1	Die Auto-Play-Funktion unterstützen	1069
21.4.2	Einen zusätzlichen Splash Screen einsetzen	1073
21.4.3	Eine Dateiverknüpfung erstellen	1075
22	App-Techniken	1081
22.1	Arbeiten mit Dateien/Verzeichnissen	1081
22.1.1	Verzeichnisinformationen auflisten	1081
22.1.2	Unterverzeichnisse auflisten	1084
22.1.3	Verzeichnisse erstellen/löschen	1085
22.1.4	Dateien auflisten	1087
22.1.5	Dateien erstellen/schreiben/lesen	1089
22.1.6	Dateien kopieren/umbenennen/löschen	1093
22.1.7	Verwenden der Dateipicker	1094
22.1.8	StorageFile-/StorageFolder-Objekte speichern	1099
22.1.9	Verwenden der Most Recently Used-Liste	1101
22.2	Datenaustausch zwischen Apps/Programmen	1102
22.2.1	Zwischenablage	1102
22.2.2	Teilen von Inhalten	1109
22.2.3	Eine App als Freigabeziel verwenden	1113
22.2.4	Zugriff auf die Kontaktliste	1114
22.3	Spezielle Oberflächenelemente	1115
22.3.1	MessageDialog	1116
22.3.2	ContentDialog	1119
22.3.3	Popup-Benachrichtigungen	1121
22.3.4	PopUp/Flyouts	1129
22.3.5	Das PopupMenu einsetzen	1132
22.3.6	Eine AppBar/CommandBar verwenden	1133
22.4	Datenbanken und Windows Store Apps	1137
22.4.1	Der Retter in der Not: SQLite!	1137
22.4.2	Verwendung/Kurzüberblick	1138
22.4.3	Installation	1139
22.4.4	Wie kommen wir zu einer neuen Datenbank?	1140
22.4.5	Wie werden die Daten manipuliert?	1144

22.5	Vertrieb der App	1146
22.5.1	Verpacken der App	1146
22.5.2	App-Installation per Skript	1148
22.6	Ein Blick auf die App-Schwachstellen	1149
22.6.1	Quellcodes im Installationsverzeichnis	1149
22.6.2	Zugriff auf den App-Datenordner	1151
22.7	Praxisbeispiele	1152
22.7.1	Ein Verzeichnis auf Änderungen überwachen	1152
22.7.2	Eine App als Freigabeziel verwenden	1154
22.7.3	ToastNotifications einfach erzeugen	1159

Anhang

A	Glossar	1167
B	Wichtige Dateiextensions	1173
	Index	1175

Download-Kapitel

LINK: <http://doko-buch.de>

Vorwort zu den Download-Kapiteln	1215
--	------

Teil IV: WPF-Anwendungen

23 Einführung in WPF	1219
23.1 Neues aus der Gerüchteküche	1220
23.2 Einführung	1220
23.2.1 Was kann eine WPF-Anwendung?	1220
23.2.2 Die eXtensible Application Markup Language	1222
23.2.3 Verbinden von XAML und C#-Code	1227
23.2.4 Zielplattformen	1232
23.2.5 Applikationstypen	1233
23.2.6 Vor- und Nachteile von WPF-Anwendungen	1234
23.2.7 Weitere Dateien im Überblick	1235
23.3 Alles beginnt mit dem Layout	1237
23.3.1 Allgemeines zum Layout	1237
23.3.2 Positionieren von Steuerelementen	1239
23.3.3 Canvas	1243
23.3.4 StackPanel	1243
23.3.5 DockPanel	1245
23.3.6 WrapPanel	1247
23.3.7 UniformGrid	1247
23.3.8 Grid	1249
23.3.9 ViewBox	1254
23.3.10 TextBlock	1255