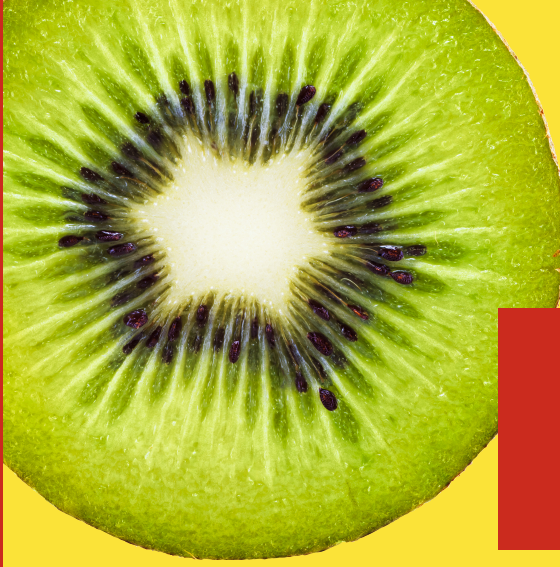




walter DOBERENZ
thomas GEWINNUS



Visual Basic 2015

**GRUNDLAGEN
PROFIWISSEN
REZEPTE**

// Visual Basic-Grundlagen
// LINQ, OOP, ADO.NET
// App-Entwicklung
// Über 150 Praxisbeispiele



EXTRA: 700 Seiten Bonuskapitel
zu WPF und Windows Forms

HANSER

Doberenz/Gewinnus

Visual Basic 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 Basic 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: 978-3-446-44380-8

E-Book-ISBN: 978-3-446-44605-2

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 VB-Programm	53
1.2.1 Vorbereitungen	53
1.2.2 Programm 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	58
1.3.2 Objekt- und ereignisorientierte Programmierung	58
1.3.3 Windows-Programmierung unter Visual Studio 2015	59
1.4 Die Entwicklungsumgebung Visual Studio 2015	61
1.4.1 Neues Projekt	61
1.4.2 Die wichtigsten Fenster	62
1.5 Microsofts .NET-Technologie	64
1.5.1 Zur Geschichte von .NET	64
1.5.2 .NET-Features und Begriffe	66
1.6 Wichtige Neuigkeiten in Visual Studio 2015	74
1.6.1 Entwicklungsumgebung	74
1.6.2 Neue VB-Sprachfeatures	74
1.6.3 Code-Editor	74
1.6.4 NET Framework 4.6	75

1.7	Praxisbeispiele	75
1.7.1	Windows-Anwendung für Einsteiger	75
1.7.2	Windows-Anwendung für fortgeschrittene Einsteiger	79
2	Einführung in Visual Basic	87
2.1	Grundbegriffe	87
2.1.1	Anweisungen	87
2.1.2	Bezeichner	88
2.1.3	Kommentare	89
2.1.4	Zeilenumbruch	90
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	93
2.2.4	Deklaration von Variablen	94
2.2.5	Typinferenz	97
2.2.6	Konstanten deklarieren	97
2.2.7	Gültigkeitsbereiche von Deklarationen	98
2.2.8	Lokale Variablen mit Dim	98
2.2.9	Lokale Variablen mit Static	99
2.2.10	Private globale Variablen	99
2.2.11	Public Variablen	100
2.3	Wichtige Datentypen im Überblick	100
2.3.1	Byte, Short, Integer, Long	100
2.3.2	Single, Double und Decimal	101
2.3.3	Char und String	101
2.3.4	Date	102
2.3.5	Boolean	103
2.3.6	Object	103
2.4	Konvertieren von Datentypen	104
2.4.1	Implizite und explizite Konvertierung	104
2.4.2	Welcher Datentyp passt zu welchem?	105
2.4.3	Konvertierungsfunktionen	106
2.4.4	CType-Funktion	107
2.4.5	Konvertieren von Strings	107
2.4.6	Die Convert-Klasse	109
2.4.7	Die Parse-Methode	109
2.4.8	Boxing und Unboxing	110

2.4.9	TryCast-Operator	111
2.4.10	Nullable Types	111
2.5	Operatoren	112
2.5.1	Arithmetische Operatoren	112
2.5.2	Zuweisungsoperatoren	113
2.5.3	Logische Operatoren	114
2.5.4	Vergleichsoperatoren	115
2.5.5	Rangfolge der Operatoren	115
2.6	Kontrollstrukturen	116
2.6.1	Verzweigungsbefehle	116
2.6.2	Schleifenanweisungen	119
2.7	Benutzerdefinierte Datentypen	120
2.7.1	Enumerationen	120
2.7.2	Strukturen	121
2.8	Nutzerdefinierte Funktionen/Prozeduren	124
2.8.1	Deklaration und Syntax	124
2.8.2	Parameterübergabe allgemein	126
2.8.3	Übergabe mit ByVal und ByRef	127
2.8.4	Optionale Parameter	128
2.8.5	Überladene Funktionen/Prozeduren	128
2.9	Praxisbeispiele	129
2.9.1	Vom PAP zum Konsolen-Programm	129
2.9.2	Vom Konsolen- zum Windows-Programm	131
2.9.3	Schleifenanweisungen kennen lernen	133
2.9.4	Methoden überladen	136
2.9.5	Eine Iterationsschleife verstehen	138
2.9.6	Anwendungen von C# nach Visual Basic portieren	141
3	OOP-Konzepte	149
3.1	Strukturierter versus objektorientierter Entwurf	149
3.1.1	Was bedeutet strukturierte Programmierung?	149
3.1.2	Was heißt objektorientierte Programmierung?	150
3.2	Grundbegriffe der OOP	151
3.2.1	Objekt, Klasse, Instanz	151
3.2.2	Kapselung und Wiederverwendbarkeit	152
3.2.3	Vererbung und Polymorphie	152
3.2.4	Sichtbarkeit von Klassen und ihren Mitgliedern	153
3.2.5	Allgemeiner Aufbau einer Klasse	154

3.3	Ein Objekt erzeugen	155
3.3.1	Referenzieren und Instanziieren	156
3.3.2	Klassische Initialisierung	157
3.3.3	Objekt-Initialisierer	157
3.3.4	Arbeiten mit dem Objekt	157
3.3.5	Zerstören des Objekts	158
3.4	OOP-Einführungsbeispiel	158
3.4.1	Vorbereitungen	158
3.4.2	Klasse definieren	159
3.4.3	Objekt erzeugen und initialisieren	160
3.4.4	Objekt verwenden	160
3.4.5	Unterstützung durch die IntelliSense	160
3.4.6	Objekt testen	161
3.4.7	Warum unsere Klasse noch nicht optimal ist	162
3.5	Eigenschaften	162
3.5.1	Eigenschaften kapseln	162
3.5.2	Eigenschaften mit Zugriffsmethoden kapseln	165
3.5.3	Lese-/Schreibschutz für Eigenschaften	166
3.5.4	Statische Eigenschaften	167
3.5.5	Selbst implementierende Eigenschaften	168
3.6	Methoden	169
3.6.1	Öffentliche und private Methoden	169
3.6.2	Überladene Methoden	170
3.6.3	Statische Methoden	171
3.7	Ereignisse	172
3.7.1	Ereignisse deklarieren	172
3.7.2	Ereignis auslösen	173
3.7.3	Ereignis auswerten	173
3.7.4	Benutzerdefinierte Ereignisse (Custom Events)	175
3.8	Arbeiten mit Konstruktor und Destruktor	178
3.8.1	Der Konstruktor erzeugt das Objekt	178
3.8.2	Bequemer geht's mit einem Objekt-Initialisierer	180
3.8.3	Destruktor und Garbage Collector räumen auf	181
3.8.4	Mit Using den Lebenszyklus des Objekts kapseln	184
3.9	Vererbung und Polymorphie	184
3.9.1	Vererbungsbeziehungen im Klassendiagramm	184
3.9.2	Überschreiben von Methoden (Method-Overriding)	186
3.9.3	Klassen implementieren	186

3.9.4	Objekte implementieren	191
3.9.5	Ausblenden von Mitgliedern durch Vererbung	192
3.9.6	Allgemeine Hinweise und Regeln zur Vererbung	194
3.9.7	Polymorphe Methoden	195
3.10	Besondere Klassen und Features	197
3.10.1	Abstrakte Klassen	197
3.10.2	Abstrakte Methoden	198
3.10.3	Versiegelte Klassen	198
3.10.4	Partielle Klassen	199
3.10.5	Die Basisklasse System.Object	201
3.10.6	Property-Accessors	202
3.10.7	Nullbedingter Operator	202
3.11	Schnittstellen (Interfaces)	203
3.11.1	Definition einer Schnittstelle	203
3.11.2	Implementieren einer Schnittstelle	204
3.11.3	Abfragen, ob eine Schnittstelle vorhanden ist	205
3.11.4	Mehrere Schnittstellen implementieren	205
3.11.5	Schnittstellenprogrammierung ist ein weites Feld	205
3.12	Praxisbeispiele	206
3.12.1	Eigenschaften sinnvoll kapseln	206
3.12.2	Eine statische Klasse anwenden	209
3.12.3	Vom fetten zum dünnen Client	211
3.12.4	Schnittstellenvererbung verstehen	220
3.12.5	Aggregation und Vererbung gegenüberstellen	224
3.12.6	Eine Klasse zur Matrizenrechnung entwickeln	230
3.12.7	Rechner für komplexe Zahlen	236
3.12.8	Formel-Rechner mit dem CodeDOM	244
3.12.9	Einen Funktionsverlauf grafisch darstellen	249
3.12.10	Sortieren mit IComparable/IComparer	253
3.12.11	Objektbäume in generischen Listen abspeichern	258
3.12.12	OOP beim Kartenspiel erlernen	264
4	Arrays, Strings, Funktionen	269
4.1	Datenfelder (Arrays)	269
4.1.1	Ein Array deklarieren	269
4.1.2	Zugriff auf Array-Elemente	270
4.1.3	Oberen Index ermitteln	270
4.1.4	Explizite Arraygrenzen	270

4.1.5	Arrays erzeugen und initialisieren	270
4.1.6	Zugriff mittels Schleife	271
4.1.7	Mehrdimensionale Arrays	272
4.1.8	Dynamische Arrays	273
4.1.9	Zuweisen von Arrays	274
4.1.10	Arrays aus Strukturvariablen	275
4.1.11	Löschen von Arrays	276
4.1.12	Eigenschaften und Methoden von Arrays	276
4.1.13	Übergabe von Arrays	279
4.2	Zeichenkettenverarbeitung	280
4.2.1	Strings zuweisen	280
4.2.2	Eigenschaften und Methoden eines Strings	280
4.2.3	Kopieren eines Strings in ein Char-Array	283
4.2.4	Wichtige (statische) Methoden der String-Klasse	283
4.2.5	Die StringBuilder-Klasse	285
4.3	Reguläre Ausdrücke	288
4.3.1	Wozu braucht man reguläre Ausdrücke?	288
4.3.2	Eine kleine Einführung	289
4.3.3	Wichtige Methoden der Klasse Regex	289
4.3.4	Kompilierte reguläre Ausdrücke	291
4.3.5	RegexOptions-Enumeration	292
4.3.6	Metazeichen (Escape-Zeichen)	293
4.3.7	Zeichenmengen (Character Sets)	294
4.3.8	Quantifizierer	295
4.3.9	Zero-Width Assertions	296
4.3.10	Gruppen	300
4.3.11	Text ersetzen	300
4.3.12	Text splitten	301
4.4	Datums- und Zeitberechnungen	302
4.4.1	Grundlegendes	302
4.4.2	Wichtige Eigenschaften von DateTime-Variablen	303
4.4.3	Wichtige Methoden von DateTime-Variablen	304
4.4.4	Wichtige Mitglieder der DateTime-Struktur	305
4.4.5	Konvertieren von Datumstrings in DateTime-Werte	306
4.4.6	Die TimeSpan-Struktur	306
4.5	Vordefinierten Funktionen	308
4.5.1	Mathematik	308
4.5.2	Datums- und Zeitfunktionen	310

4.6	Zahlen formatieren	312
4.6.1	Die ToString-Methode	313
4.6.2	Die Format-Methode	314
4.6.3	Stringinterpolation	316
4.7	Praxisbeispiele	316
4.7.1	Zeichenketten verarbeiten	316
4.7.2	Zeichenketten mittels StringBuilder addieren	319
4.7.3	Reguläre Ausdrücke testen	323
4.7.4	Fehler bei mathematischen Operationen behandeln	325
4.7.5	Methodenaufrufe mit Array-Parametern	328
4.7.6	String in Array kopieren und umgekehrt	331
4.7.7	Ein Byte-Array in einen String konvertieren	333
4.7.8	Strukturvariablen in Arrays einsetzen	335
5	Weitere Sprachfeatures	339
5.1	Namespaces (Namensräume)	339
5.1.1	Ein kleiner Überblick	339
5.1.2	Die Imports-Anweisung	341
5.1.3	Namespace-Alias	341
5.1.4	Namespaces in Projekteigenschaften	342
5.1.5	Namespace Alias Qualifizierer	343
5.1.6	Eigene Namespaces einrichten	343
5.2	Überladen von Operatoren	344
5.2.1	Syntaxregeln	344
5.2.2	Praktische Anwendung	345
5.2.3	Konvertierungsoperatoren überladen	346
5.3	Auflistungen (Collections)	347
5.3.1	Beziehungen zwischen den Schnittstellen	347
5.3.2	IEnumerable	348
5.3.3	ICollection	349
5.3.4	ICollection	349
5.3.5	Iteratoren	349
5.3.6	Die ArrayList-Collection	350
5.3.7	Die Hashtable	351
5.4	Generische Datentypen	352
5.4.1	Wie es früher einmal war	352
5.4.2	Typsicherheit durch Generics	354
5.4.3	List-Collection ersetzt ArrayList	355

5.4.4	Über die Vorzüge generischer Collections	356
5.4.5	Typbeschränkungen durch Constraints	357
5.4.6	Collection-Initialisierer	358
5.4.7	Generische Methoden	358
5.5	Delegates	359
5.5.1	Delegates sind Methodenzeiger	359
5.5.2	Delegate-Typ definieren	360
5.5.3	Delegate-Objekt erzeugen	361
5.5.4	Delegates vereinfacht instanziiieren	361
5.5.5	Relaxed Delegates	362
5.5.6	Anonyme Methoden	362
5.5.7	Lambda-Ausdrücke	363
5.5.8	Lambda-Ausdrücke in der Task Parallel Library	364
5.6	Dynamische Programmierung	366
5.6.1	Wozu dynamische Programmierung?	366
5.6.2	Das Prinzip der dynamischen Programmierung	366
5.6.3	Kovarianz und Kontravarianz	370
5.7	Weitere Datentypen	371
5.7.1	BigInteger	371
5.7.2	Complex	373
5.7.3	Tuple(Of T)	374
5.7.4	SortedSet(Of T)	374
5.8	Praxisbeispiele	376
5.8.1	ArrayList versus generische List	376
5.8.2	Delegates und Lambda Expressions	379
5.8.3	Mit dynamischem Objekt eine Datei durchsuchen	382
6	Einführung in LINQ	387
6.1	LINQ-Grundlagen	387
6.1.1	Die LINQ-Architektur	387
6.1.2	LINQ-Implementierungen	388
6.1.3	Anonyme Typen	388
6.1.4	Erweiterungsmethoden	390
6.2	Abfragen mit LINQ to Objects	391
6.2.1	Grundlegendes zur LINQ-Syntax	391
6.2.2	Zwei alternative Schreibweisen von LINQ-Abfragen	392
6.2.3	Übersicht der wichtigsten Abfrage-Operatoren	394

6.3	LINQ-Abfragen im Detail	395
6.3.1	Die Projektionsoperatoren Select und SelectMany	396
6.3.2	Der Restriktionsoperator Where	398
6.3.3	Die Sortierungsoperatoren OrderBy und ThenBy	398
6.3.4	Der Gruppierungsoperator GroupBy	400
6.3.5	Verknüpfen mit Join	401
6.3.6	Aggregat-Operatoren	402
6.3.7	Verzögertes Ausführen von LINQ-Abfragen	404
6.3.8	Konvertierungsmethoden	405
6.3.9	Abfragen mit PLINQ	405
6.4	Praxisbeispiele	408
6.4.1	Die Syntax von LINQ-Abfragen verstehen	408
6.4.2	Aggregat-Abfragen mit LINQ	411
6.4.3	LINQ im Schnelldurchgang erlernen	413
6.4.4	Strings mit LINQ abfragen und filtern	416
6.4.5	Duplikate aus einer Liste oder einem Array entfernen	417
6.4.6	Arrays mit LINQ initialisieren	420
6.4.7	Arrays per LINQ mit Zufallszahlen füllen	422
6.4.8	Einen String mit Wiederholmuster erzeugen	423
6.4.9	Mit LINQ Zahlen und Strings sortieren	425
6.4.10	Mit LINQ Collections von Objekten sortieren	426
6.4.11	Ergebnisse von LINQ-Abfragen in ein Array kopieren	428

Teil II: Technologien

7	Zugriff auf das Dateisystem	431
7.1	Grundlagen	431
7.1.1	Klassen für Verzeichnis- und Dateioperationen	432
7.1.2	Statische versus Instanzen-Klasse	432
7.2	Übersichten	433
7.2.1	Methoden der Directory-Klasse	433
7.2.2	Methoden eines DirectoryInfo-Objekts	434
7.2.3	Eigenschaften eines DirectoryInfo-Objekts	434
7.2.4	Methoden der File-Klasse	434
7.2.5	Methoden eines FileInfo-Objekts	435
7.2.6	Eigenschaften eines FileInfo-Objekts	436

7.3	Operationen auf Verzeichnisebene	436
7.3.1	Existenz eines Verzeichnisses/einer Datei feststellen	436
7.3.2	Verzeichnisse erzeugen und löschen	437
7.3.3	Verzeichnisse verschieben und umbenennen	438
7.3.4	Aktuelles Verzeichnis bestimmen	438
7.3.5	Unterverzeichnisse ermitteln	438
7.3.6	Alle Laufwerke ermitteln	439
7.3.7	Dateien kopieren und verschieben	440
7.3.8	Dateien umbenennen	441
7.3.9	Dateiattribute feststellen	441
7.3.10	Verzeichnis einer Datei ermitteln	443
7.3.11	Alle im Verzeichnis enthaltene Dateien ermitteln	443
7.3.12	Dateien und Unterverzeichnisse ermitteln	443
7.4	Zugriffsberechtigungen	444
7.4.1	ACL und ACE	444
7.4.2	SetAccessControl-Methode	445
7.4.3	Zugriffsrechte anzeigen	445
7.5	Weitere wichtige Klassen	446
7.5.1	Die Path-Klasse	446
7.5.2	Die Klasse FileSystemWatcher	447
7.6	Datei- und Verzeichnisdialoge	449
7.6.1	OpenFileDialog und SaveFileDialog	449
7.6.2	FolderBrowserDialog	451
7.7	Praxisbeispiele	452
7.7.1	Infos über Verzeichnisse und Dateien gewinnen	452
7.7.2	Die Verzeichnisstruktur in eine TreeView einlesen	455
7.7.3	Mit LINQ und RegEx Verzeichnisbäume durchsuchen	457
8	Dateien lesen und schreiben	463
8.1	Grundprinzip der Datenpersistenz	463
8.1.1	Dateien und Streams	463
8.1.2	Die wichtigsten Klassen	464
8.1.3	Erzeugen eines Streams	465
8.2	Dateiparameter	465
8.2.1	FileAccess	465
8.2.2	FileMode	465
8.2.3	FileShare	466

8.3	Textdateien	466
8.3.1	Eine Textdatei beschreiben bzw. neu anlegen	466
8.3.2	Eine Textdatei lesen	468
8.4	Binärdateien	470
8.4.1	Lese-/Schreibzugriff	470
8.4.2	Die Methoden ReadAllBytes und WriteAllBytes	470
8.4.3	BinaryReader/BinaryWriter erzeugen	471
8.5	Sequenzielle Dateien	471
8.5.1	Lesen und Schreiben von strukturierten Daten	471
8.5.2	Serialisieren von Objekten	472
8.6	Dateien verschlüsseln und komprimieren	473
8.6.1	Das Methodenpärchen Encrypt-/Decrypt	474
8.6.2	Verschlüsseln unter Windows Vista/7/8/10	474
8.6.3	Verschlüsseln mit der CryptoStream-Klasse	475
8.6.4	Dateien komprimieren	476
8.7	Memory Mapped Files	477
8.7.1	Grundprinzip	477
8.7.2	Erzeugen eines MMF	478
8.7.3	Erstellen eines Map View	478
8.8	Praxisbeispiele	479
8.8.1	Auf eine Textdatei zugreifen	479
8.8.2	Einen Objektbaum speichern	483
8.8.3	Ein Memory Mapped File (MMF) verwenden	490
8.8.4	Hex-Dezimal-Bytes-Konverter	492
8.8.5	Eine Datei verschlüsseln	496
8.8.6	Eine Datei komprimieren	499
8.8.7	Echte ZIP-Dateien erstellen	501
8.8.8	PDFs erstellen/exportieren	502
8.8.9	Eine CSV-Datei erstellen	506
8.8.10	Eine CSV-Datei mit LINQ lesen und auswerten	509
8.8.11	Einen korrekten Dateinamen erzeugen	511
9	Asynchrone Programmierung	513
9.1	Übersicht	513
9.1.1	Multitasking versus Multithreading	514
9.1.2	Deadlocks	515
9.1.3	Racing	515

9.2	Programmieren mit Threads	517
9.2.1	Einführungsbeispiel	517
9.2.2	Wichtige Thread-Methoden	518
9.2.3	Wichtige Thread-Eigenschaften	520
9.2.4	Einsatz der ThreadPool-Klasse	521
9.3	Sperrmechanismen	523
9.3.1	Threading ohne SyncLock	523
9.3.2	Threading mit SyncLock	524
9.3.3	Die Monitor-Klasse	527
9.3.4	Mutex	530
9.3.5	Methoden für die parallele Ausführung sperren	531
9.3.6	Semaphore	532
9.4	Interaktion mit der Programmoberfläche	533
9.4.1	Die Werkzeuge	534
9.4.2	Einzelne Steuerelemente mit Invoke aktualisieren	534
9.4.3	Mehrere Steuerelemente aktualisieren	535
9.4.4	Ist ein Invoke-Aufruf nötig?	536
9.4.5	Und was ist mit WPF?	536
9.5	Timer-Threads	538
9.6	Die BackgroundWorker-Komponente	539
9.7	Asynchrone Programmier-Entwurfsmuster	542
9.7.1	Kurzübersicht	542
9.7.2	Polling	543
9.7.3	Callback verwenden	544
9.7.4	Callback mit Parameterübergabe verwenden	545
9.7.5	Callback mit Zugriff auf die Programm-Oberfläche	546
9.8	Asynchroner Aufruf beliebiger Methoden	547
9.8.1	Die Beispielklasse	547
9.8.2	Asynchroner Aufruf ohne Callback	549
9.8.3	Asynchroner Aufruf mit Callback und Anzeigefunktion	549
9.8.4	Aufruf mit Rückgabewerten (per Eigenschaft)	550
9.8.5	Aufruf mit Rückgabewerten (per EndInvoke)	551
9.9	Es geht auch einfacher – Async und Await	552
9.9.1	Der Weg von synchron zu asynchron	552
9.9.2	Achtung: Fehlerquellen!	554
9.9.3	Eigene asynchrone Methoden entwickeln	556

9.10	Praxisbeispiele	558
9.10.1	Spieltrieb & Multithreading erleben	558
9.10.2	Prozess- und Thread-Informationen gewinnen	570
9.10.3	Ein externes Programm starten	575
10	Die Task Parallel Library	579
10.1	Überblick	579
10.1.1	Parallel-Programmierung	579
10.1.2	Möglichkeiten der TPL	582
10.1.3	Der CLR-Threadpool	582
10.2	Parallele Verarbeitung mit Parallel.Invoke	583
10.2.1	Aufrufvarianten	584
10.2.2	Einschränkungen	585
10.3	Verwendung von Parallel.For	585
10.3.1	Abbrechen der Verarbeitung	587
10.3.2	Auswerten des Verarbeitungsstatus	588
10.3.3	Und was ist mit anderen Iterator-Schrittweiten?	588
10.4	Collections mit Parallel.ForEach verarbeiten	589
10.5	Die Task-Klasse	590
10.5.1	Einen Task erzeugen	590
10.5.2	Task starten	591
10.5.3	Datenübergabe an den Task	592
10.5.4	Wie warte ich auf das Taskende?	593
10.5.5	Tasks mit Rückgabewerten	595
10.5.6	Die Verarbeitung abbrechen	598
10.5.7	Fehlerbehandlung	602
10.5.8	Weitere Eigenschaften	602
10.6	Zugriff auf das Userinterface	604
10.6.1	Task-Ende und Zugriff auf die Oberfläche	604
10.6.2	Zugriff auf das UI aus dem Task heraus	605
10.7	Weitere Datenstrukturen im Überblick	607
10.7.1	Threadsichere Collections	607
10.7.2	Primitive für die Threadsynchronisation	608
10.8	Parallel LINQ (PLINQ)	608
10.9	Praxisbeispiel: Spieltrieb – Version 2	609
10.9.1	Aufgabenstellung	609
10.9.2	Global-Klasse	609
10.9.3	Controller	610

10.9.4	LKWs	612
10.9.5	Schiff-Klasse	613
10.9.6	Oberfläche	615
11	Fehlersuche und Behandlung	617
11.1	Der Debugger	617
11.1.1	Allgemeine Beschreibung	617
11.1.2	Die wichtigsten Fenster	618
11.1.3	Debugging-Optionen	621
11.1.4	Praktisches Debugging am Beispiel	623
11.2	Arbeiten mit Debug und Trace	627
11.2.1	Wichtige Methoden von Debug und Trace	627
11.2.2	Besonderheiten der Trace-Klasse	630
11.2.3	TraceListener-Objekte	631
11.3	Caller Information	634
11.3.1	Attribute	634
11.3.2	Anwendung	634
11.4	Fehlerbehandlung	635
11.4.1	Anweisungen zur Fehlerbehandlung	635
11.4.2	Try-Catch	635
11.4.3	Try-Finally	640
11.4.4	Das Standardverhalten bei Ausnahmen festlegen	642
11.4.5	Die Exception-Klasse	643
11.4.6	Fehler/Ausnahmen auslösen	643
11.4.7	Eigene Fehlerklassen	644
11.4.8	Exceptionhandling zur Entwurfszeit	646
11.4.9	Code Contracts	646
12	XML in Theorie und Praxis	649
12.1	XML – etwas Theorie	649
12.1.1	Übersicht	649
12.1.2	Der XML-Grundaufbau	652
12.1.3	Wohlgeformte Dokumente	653
12.1.4	Processing Instructions (PI)	656
12.1.5	Elemente und Attribute	656
12.1.6	Verwendbare Zeichensätze	658

12.2	XSD-Schemas	660
12.2.1	XSD-Schemas und ADO.NET	660
12.2.2	XML-Schemas in Visual Studio analysieren	662
12.2.3	XML-Datei mit XSD-Schema erzeugen	665
12.2.4	XSD-Schema aus einer XML-Datei erzeugen	666
12.3	XML-Integration in Visual Basic	667
12.3.1	XML-Literale	667
12.3.2	Einfaches Navigieren durch späte Bindung	670
12.3.3	Die LINQ to XML-API	672
12.3.4	Neue XML-Dokumente erzeugen	673
12.3.5	Laden und Sichern von XML-Dokumenten	675
12.3.6	Navigieren in XML-Daten	677
12.3.7	Auswählen und Filtern	679
12.3.8	Manipulieren der XML-Daten	679
12.3.9	XML-Dokumente transformieren	681
12.4	Verwendung des DOM unter .NET	684
12.4.1	Übersicht	684
12.4.2	DOM-Integration in Visual Basic	685
12.4.3	Laden von Dokumenten	685
12.4.4	Erzeugen von XML-Dokumenten	686
12.4.5	Auslesen von XML-Dateien	688
12.4.6	Direktzugriff auf einzelne Elemente	689
12.4.7	Einfügen von Informationen	690
12.4.8	Suchen in den Baumzweigen	692
12.5	Weitere Möglichkeiten der XML-Verarbeitung	696
12.5.1	Die relationale Sicht mit XmlDataDocument	696
12.5.2	XML-Daten aus Objektstrukturen erzeugen	699
12.5.3	Schnelles Suchen in XML-Daten mit XPathNavigator	702
12.5.4	Schnelles Auslesen von XML-Daten mit XmlReader	705
12.5.5	Erzeugen von XML-Daten mit XmlWriter	707
12.5.6	XML transformieren mit XSLT	709
12.6	Praxisbeispiele	711
12.6.1	Mit dem DOM in XML-Dokumenten navigieren	711
12.6.2	XML-Daten in eine TreeView einlesen	714
12.6.3	DataSets in XML-Strings konvertieren	718
12.6.4	In Dokumenten mit dem XPathNavigator navigieren	722

13	Einführung in ADO.NET	727
13.1	Eine kleine Übersicht	727
13.1.1	Die ADO.NET-Klassenhierarchie	727
13.1.2	Die Klassen der Datenprovider	728
13.1.3	Das Zusammenspiel der ADO.NET-Klassen	731
13.2	Das Connection-Objekt	732
13.2.1	Allgemeiner Aufbau	732
13.2.2	OleDbConnection	732
13.2.3	Schließen einer Verbindung	734
13.2.4	Eigenschaften des Connection-Objekts	734
13.2.5	Methoden des Connection-Objekts	736
13.2.6	DerConnectionStringBuilder	737
13.3	Das Command-Objekt	738
13.3.1	Erzeugen und Anwenden eines Command-Objekts	738
13.3.2	Erzeugen mittels CreateCommand-Methode	739
13.3.3	Eigenschaften des Command-Objekts	739
13.3.4	Methoden des Command-Objekts	741
13.3.5	Freigabe von Connection- und Command-Objekten	742
13.4	Parameter-Objekte	744
13.4.1	Erzeugen und Anwenden eines Parameter-Objekts	744
13.4.2	Eigenschaften des Parameter-Objekts	744
13.5	Das CommandBuilder-Objekt	745
13.5.1	Erzeugen	745
13.5.2	Anwenden	746
13.6	Das DataReader-Objekt	746
13.6.1	DataReader erzeugen	747
13.6.2	Daten lesen	747
13.6.3	Eigenschaften des DataReaders	748
13.6.4	Methoden des DataReaders	748
13.7	Das DataAdapter-Objekt	749
13.7.1	DataAdapter erzeugen	749
13.7.2	Command-Eigenschaften	750
13.7.3	Fill-Methode	751
13.7.4	Update-Methode	752
13.8	Praxisbeispiele	753
13.8.1	Wichtige ADO.NET-Objekte im Einsatz	753
13.8.2	Eine Aktionsabfrage ausführen	755
13.8.3	Eine Auswahlabfrage aufrufen	757

13.8.4	Die Datenbank aktualisieren	759
13.8.5	DenConnectionString speichern	762
14	Das DataSet	765
14.1	Grundlegende Features des DataSets	765
14.1.1	Die Objekthierarchie	766
14.1.2	Die wichtigsten Klassen	766
14.1.3	Erzeugen eines DataSets	767
14.2	Das DataTable-Objekt	769
14.2.1	DataTable erzeugen	769
14.2.2	Spalten hinzufügen	769
14.2.3	Zeilen zur DataTable hinzufügen	770
14.2.4	Auf den Inhalt einer DataTable zugreifen	771
14.3	Die DataView	773
14.3.1	Erzeugen eines DataView	773
14.3.2	Sortieren und Filtern von Datensätzen	773
14.3.3	Suchen von Datensätzen	774
14.4	Typisierte DataSets	774
14.4.1	Ein typisiertes DataSet erzeugen	775
14.4.2	Das Konzept der Datenquellen	776
14.4.3	Typisierte DataSets und TableAdapter	777
14.5	Die Qual der Wahl	778
14.5.1	DataReader – der schnelle Lesezugriff	779
14.5.2	DataSet – die Datenbank im Hauptspeicher	779
14.5.3	Objektrelationales Mapping – die Zukunft?	780
14.6	Praxisbeispiele	781
14.6.1	Im DataView sortieren und filtern	781
14.6.2	Suche nach Datensätzen	783
14.6.3	Ein DataSet in einen XML-String serialisieren	784
14.6.4	Untypisierte in typisierte DataSets konvertieren	789
14.6.5	Eine LINQ to SQL-Abfrage ausführen	794
15	Verteilen von Anwendungen	799
15.1	ClickOnce-Deployment	800
15.1.1	Übersicht/Einschränkungen	800
15.1.2	Die Vorgehensweise	801
15.1.3	Ort der Veröffentlichung	801
15.1.4	Anwendungsdateien	802

15.1.5	Erforderliche Komponenten	802
15.1.6	Aktualisierungen	803
15.1.7	Veröffentlichungsoptionen	804
15.1.8	Veröffentlichen	805
15.1.9	Verzeichnisstruktur	805
15.1.10	Der Webpublishing-Assistent	807
15.1.11	Neue Versionen erstellen	808
15.2	InstallShield	808
15.2.1	Installation	808
15.2.2	Aktivieren	809
15.2.3	Ein neues Setup-Projekt	809
15.2.4	Finaler Test	817
15.3	Hilfdateien programmieren	817
15.3.1	Der HTML Help Workshop	818
15.3.2	Bedienung am Beispiel	819
15.3.3	Hilfdateien in die VB-Anwendung einbinden	821
15.3.4	Eine alternative Hilfe-IDE verwenden	825
16	Weitere Techniken	827
16.1	Zugriff auf die Zwischenablage	827
16.1.1	Das Clipboard-Objekt	827
16.1.2	Zwischenablage-Funktionen für Textboxen	829
16.2	Arbeiten mit der Registry	829
16.2.1	Allgemeines	830
16.2.2	Registry-Unterstützung in .NET	831
16.3	.NET-Reflection	833
16.3.1	Übersicht	833
16.3.2	Assembly laden	833
16.3.3	Mittels GetType und Type Informationen sammeln	834
16.3.4	Dynamisches Laden von Assemblies	836
16.4	Praxisbeispiele	838
16.4.1	Zugriff auf die Registry	838
16.4.2	Dateiverknüpfungen erzeugen	840
16.4.3	Die Zwischenablage überwachen und anzeigen	842
16.4.4	Die WIA-Library kennenlernen	845
16.4.5	Auf eine Webcam zugreifen	857
16.4.6	Auf den Scanner zugreifen	859
16.4.7	OpenOffice.org Writer per OLE steuern	863

16.4.8	Nutzer und Gruppen des Systems ermitteln	871
16.4.9	Testen, ob Nutzer in einer Gruppe enthalten ist	872
16.4.10	Testen, ob der Nutzer ein Administrator ist	874
16.4.11	Die IP-Adressen des Computers bestimmen	875
16.4.12	Die IP-Adresse über den Hostnamen bestimmen	876
16.4.13	Diverse Systeminformationen ermitteln	877
16.4.14	Sound per MCI aufnehmen	886
16.4.15	Mikrofonpegel anzeigen	889
16.4.16	Pegeldiagramm aufzeichnen	891
16.4.17	Sound-und Video-Dateien per MCI abspielen	895
17	Konsolenanwendungen	903
17.1	Grundaufbau/Konzepte	903
17.1.1	Unser Hauptprogramm – Module1.vb	904
17.1.2	Rückgabe eines Fehlerstatus	905
17.1.3	Parameterübergabe	906
17.1.4	Zugriff auf die Umgebungsvariablen	907
17.2	Die Kommandozentrale: System.Console	908
17.2.1	Eigenschaften	908
17.2.2	Methoden/Ereignisse	909
17.2.3	Textausgaben	910
17.2.4	Farbangaben	911
17.2.5	Tastaturabfragen	912
17.2.6	Arbeiten mit Streamdaten	913
17.3	Praxisbeispiel: Farbige Konsolenanwendung	914

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 Store 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	Zielpattformen	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	968
19.5.1	Seitennavigation und Parameterübergabe	968
19.5.2	Die Fensterkopfzeile anpassen	970
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	984
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	996
20.3	Listendarstellungen	998
20.3.1	ComboBox, ListBox	998
20.3.2	ListView	1002
20.3.3	GridView	1004
20.3.4	FlipView	1006
20.4	Sonstige Controls	1008
20.4.1	CaptureElement	1008
20.4.2	MediaElement	1009
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	1020
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.vb	1038
21.1.3	Verweise	1040
21.1.4	App.xaml und App.xaml.vb	1040
21.1.5	Package.appxmanifest	1041
21.1.6	Application1_TemporaryKey.pfx	1046
21.1.7	MainPage.xaml & MainPage.xaml.vb	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	1052
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?	1059
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	1064
21.3.8	Daten in der Cloud ablegen/laden (Roaming)	1066
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	1086
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	1095
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	1112
22.2.4	Zugriff auf die Kontaktliste	1113
22.3	Spezielle Oberflächenelemente	1115
22.3.1	MessageDialog	1115
22.3.2	ContentDialog	1118
22.3.3	Popup-Benachrichtigungen	1120
22.3.4	PopUp/Flyouts	1127
22.3.5	Das PopupMenu einsetzen	1131
22.3.6	Eine AppBar 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	1137
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	1145
22.5.1	Verpacken der App	1145
22.5.2	App-Installation per Skript	1147

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	1151
22.7.1	Ein Verzeichnis auf Änderungen überwachen	1151
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	Einführung	1220
23.1.1	Was kann eine WPF-Anwendung?	1220
23.1.2	Die eXtensible Application Markup Language	1222
23.1.3	Verbinden von XAML und Basic-Code	1227
23.1.4	Zielpattformen	1232
23.1.5	Applikationstypen	1233
23.1.6	Vorteile und Nachteile von WPF-Anwendungen	1234
23.1.7	Weitere Dateien im Überblick	1235
23.2	Alles beginnt mit dem Layout	1238
23.2.1	Allgemeines zum Layout	1238
23.2.2	Positionieren von Steuerelementen	1240
23.2.3	Canvas	1243
23.2.4	StackPanel	1244
23.2.5	DockPanel	1246
23.2.6	WrapPanel	1248
23.2.7	UniformGrid	1248
23.2.8	Grid	1250
23.2.9	ViewBox	1254
23.2.10	TextBlock	1255
23.3	Das WPF-Programm	1258
23.3.1	Die Application-Klasse	1259
23.3.2	Das Startobjekt festlegen	1259
23.3.3	Kommandozeilenparameter verarbeiten	1261