

Übersicht

Einleitung	17
1 Das erste Visual Basic .NET-Beispiel	31
2 Die .NET-Philosophie	47
3 Visual Basic .NET-Einmaleins	83
4 Kontrollstrukturen	113
5 Funktionen und Prozeduren	137
6 Klassen	147
7 Zeichenketten (Strings)	173
8 Arrays (Felder)	197
9 Dateizugriffe und Ausnahmen	227
10 Fortgeschrittenere Programmelemente in Visual Basic .NET	259
11 OOP für (etwas) Fortgeschrittene	297
12 XML, was sonst?	335
13 Visual Basic .NET und das Internet	367
14 Windows Forms, Controls und Visual Studio .NET	387
Anhang A: .NET von A bis Z	413
Anhang B: Visual Basic .NET-Referenz	417
Anhang C: Überblick über das .NET-Framework-SDK	427
Anhang D: Antworten	433
Stichwortverzeichnis	473

Inhaltsverzeichnis

Einleitung	17
1 Das erste Visual Basic .NET-Beispiel	31
1.1 Auch .NET kann Hallo sagen	31
1.2 Der allgemeine Aufbau einer Konsolenanwendung	34
1.2.1 Der allgemeine Programmrahmen einer Konsolenanwendung	35
1.3 Ein paar Spezialitäten in .NET	36
1.3.1 Dateizugriffe in Visual Basic .NET	36
1.3.2 XML-Dateien lesen und auswerten	37
1.3.3 Herunterladen einer HTML-Datei von einem Webserver	39
1.3.4 Zugriff auf die Metadaten eines Assembly	40
1.3.5 Aufruf einer Windows-API-Funktion	42
1.4 Die wichtigsten Compiler-Optionen	43
1.5 Zusammenfassung	45
1.6 F&A	46
2 Die .NET-Philosophie	47
2.1 Was ist .NET?	49
2.1.1 Die Rolle der Common Language Runtime (CLR)	50
2.1.2 IL als die neue Maschinensprache	52
2.2 .NET im Gesamtüberblick	54
2.3 Die Datentypen des Common Type System (CTS)	56
2.4 Die .NET-Basisklassen	58

2.4.1	Die Rolle der Namensräume	58
2.4.2	Die .NET-Basisklassen in der Praxis	60
2.5	Die Rolle der Assemblies	62
2.5.1	Starke Namen, schwache Namen	65
2.6	Vorteile für Programmierer	69
2.6.1	Versionierung	70
2.6.2	Sicherheit	71
2.6.3	Vereinfachung durch Laufzeitdienste	72
2.6.4	Vereinheitlichung	72
2.6.5	Leistungsfähigere Programmiersprachen	72
2.6.6	Durch und durch objektorientiert	73
2.6.7	Zusammenarbeit zwischen Programmen	73
2.6.8	Realisierung verteilter Anwendungen	76
2.6.9	Mehr Spaß	76
2.7	Weitere Bestandteile von .NET	77
2.7.1	C# und Visual Basic .NET	77
2.7.2	ASP .NET	78
2.7.3	Webdienste	78
2.7.4	ADO .NET	79
2.7.5	Windows Forms	79
2.8	Die Common Language Infrastructure (CLI) für Linux & Co.	79
2.9	Zusammenfassung	80
2.10	F&A	81
3	Visual Basic .NET-Einmaleins	83
3.1	Kleine Merkzettel für das Programm: die Variablen	84
3.1.1	Variablen mit einem Anfangswert belegen	84
3.1.2	Mehrere Variablen in einem Befehl deklarieren	84
3.1.3	Mehrere Variablen mit unterschiedlichen Datentypen in einem Befehl deklarieren	84
3.1.4	Weitere Deklarationsbefehle	85
3.2	Die Merkmale einer Variablen	85
3.2.1	Formal und trotzdem unverzichtbar: der Datentyp	85
3.2.2	Typisch Basic – der Datentyp ist freiwillig	88
3.2.3	Die Rolle des Gültigkeitsbereichs: Wo werden Variablen in einem Programm deklariert?	89
3.2.4	Die Rolle der Lebensdauer einer Variablen	92
3.3	Charakterfest und beständig: die Konstanten	93
3.3.1	Auch Zahlenwerte sind Konstanten	94
3.3.2	Zeichenkettenkonstanten	94

3.4	Regeln für Bezeichner	95
3.5	Spezialisten fürs Rechnen: die Operatoren	95
3.5.1	Ein kleiner Ausflug in die Mathe-Klasse	96
3.5.2	Zuweisungen	97
3.5.3	Alles genau nach Vorschrift: die Operatorreihenfolge	98
3.5.4	Kombiausdrücke: Zuweisung und Operator in einem	98
3.6	Wahr oder falsch? – die Welt der logischen Verknüpfungen	99
3.6.1	Die Und-Verknüpfung mit dem And-Operator	100
3.6.2	Die Oder-Verknüpfung mit dem Or-Operator	101
3.6.3	Die Exklusiv-Oder-Verknüpfung mit dem XOR-Operator	101
3.6.4	Die Negation mit dem Not-Operator	101
3.6.5	Die kombinierten logischen Operatoren AndAlso und OrAlso	101
3.7	Eingaben von der Tastatur	103
3.8	Ausgaben auf dem Bildschirm	103
3.8.1	Formatierte Ausgaben	104
3.8.2	Ausgaben über die MsgBox-Funktion	106
3.8.3	Die InputBox-Funktion zur Eingabe	107
3.9	Der allgemeine Aufbau eines Visual Basic .NET-Programms	107
3.9.1	Die Rolle der Visual Basic .NET-Module	108
3.10	F&A	110
4	Kontrollstrukturen	113
4.1	Das Prinzip der Entscheidungen	114
4.2	Wenn etwas so ist, dann tue das – der If-Befehl	115
4.2.1	Vergleichsoperatoren	117
4.2.2	Bedingungen kombinieren	117
4.3	Es gibt immer eine Alternative – der Else-Befehl	118
4.4	Der ElseIf-Befehl als Abkürzung	119
4.5	Mehrfachentscheidungen mit dem Select Case-Befehl	121
4.5.1	Einen Case-Zweig vorzeitig verlassen	125
4.6	Wiederholungen mit System – der For Next-Befehl	125
4.6.1	Zählschleifen mit For Next	125
4.6.2	Wiederholungen mit Abbruchbedingung – der Do Loop-Befehl	129
4.6.3	Eine Alternative, die eigentlich keine ist – der While-Befehl	130
4.7	Schleifen mit Notausgang – der Exit-Befehl	131
4.8	Wie »schnell« ist Visual Basic .NET?	133
4.9	Zusammenfassung	134
4.10	F&A	134

5	Funktionen und Prozeduren	137
5.1	Die Definition von Funktionen und Prozeduren	138
5.1.1	Der allgemeine Aufbau einer Funktion	138
5.1.2	Der allgemeine Aufbau einer Prozedur	139
5.1.3	Die Rolle des Rückgabewerts	139
5.2	Parameterübergabe bei Funktionen	139
5.2.1	Benennung von Funktionsparametern	140
5.2.2	Parameterübergabe als Wert	141
5.2.3	Optionale Parameter	141
5.2.4	Funktionen mit einer variablen Anzahl an Parametern	142
5.3	Die Rolle des Funktionsnamens als lokale Variable	142
5.4	Rekursive Funktionen	143
5.5	Prozedur mit »Notausgang«	144
5.6	Zusammenfassung	144
5.7	F&A	145
6	Klassen	147
6.1	Klassen definieren	148
6.2	Klassen instanzieren	149
6.2.1	Vermeiden Sie späte Bindung	149
6.3	Felder, Eigenschaften und Methoden implementieren	150
6.3.1	Felder implementieren	150
6.3.2	Eigenschaften implementieren	151
6.3.3	Methoden implementieren	154
6.4	Objekte benutzen	155
6.5	Die Bedeutung von Me	156
6.6	Objekte zuweisen und vergleichen	158
6.6.1	Objekte vergleichen	159
6.6.2	Feststellen, ob eine Instanz von einer bestimmten Klasse abstammt	160
6.7	Gemeinsam oder Instanz?	161
6.8	Objekt oder Klasse?	163
6.9	Die Rolle des Konstruktors	164
6.10	Eigenschaften und Methoden überladen	165
6.11	Klassen und der allgemeine Programmaufbau	166
6.12	Klassen und wozu sie gut sind	167
6.13	Die .NET-Basisklassen	168
6.13.1	Die Rolle des Imports-Befehl	170
6.14	Zusammenfassung	171
6.15	F&A	171

7	Zeichenketten (Strings)	173
7.1	Die Definition einer String-Variablen	174
7.2	Überblick über die String-Klasse	175
7.3	String-Operationen in der Praxis	177
7.3.1	Strings zusammenfügen	177
7.3.2	Strings vergleichen	177
7.3.3	Einen String in Elemente zerlegen	179
7.3.4	Einen String durchsuchen	180
7.3.5	Elemente in einem String austauschen	180
7.3.6	Elemente zu einem String zusammenfügen	181
7.3.7	Elemente aus einem String entfernen	181
7.3.8	Elemente aus einem String zurückgegeben	182
7.3.9	Feststellen, ob ein String mit einem bestimmten Teilstring beginnt oder endet	182
7.3.10	Strings mit Leerzeichen füllen	182
7.3.11	Strings »zurechtschneiden«	182
7.3.12	Die Länge einer Zeichenkette feststellen	183
7.3.13	Strings ohne Wert	183
7.3.14	Ein Beispiel für die wichtigsten String-Operationen	184
7.4	Die StringBuilder-Klasse für den effektiven Umgang mit Strings	189
7.5	Spezialitäten beim Umgang mit der String-Klasse	190
7.5.1	Strings anlegen einmal ganz anders	191
7.5.2	Noch einmal Strings zusammensetzen	192
7.5.3	Wie passt das alles zusammen?	192
7.5.4	Weitere Fakten über Strings bei .NET	193
7.6	Zusammenfassung	193
7.7	F&A	194
8	Arrays (Felder)	197
8.1	Einer für viele – Feldvariablen stellen sich vor	198
8.1.1	Feldvariablen mit ihrer Deklaration initialisieren	199
8.1.2	Wachsen oder Schrumpfen kein Problem – der ReDim-Befehl	200
8.1.3	Bitte nichts anfassen – das Schlüsselwort Preserve	200
8.1.4	Feststellen der Anzahl der Elemente	201
8.1.5	Feststellen der Obergrenze eines Feldes	202
8.2	Die nächste Dimension – mehrdimensionale Felder	202
8.3	Die Array-Klasse stellt sich vor	212
8.3.1	Instanzenmethoden und Eigenschaften der Array-Klasse	215

8.4	Spezialitäten beim Umgang mit Arrays	215
8.4.1	Die Suche in einem Array	216
8.4.2	Arrays gegenseitig zuweisen	217
8.4.3	Felder löschen	218
8.4.4	Arrays nach anderen Kriterien sortieren	218
8.4.5	Arrays als Rückgabewerte von Funktionen	219
8.4.6	Arrays mit einer anderen Untergrenze als 0	220
8.5	In den »Tiefen« der Basisklassen – die <i>ArrayList</i> -Klasse als Spezialist mit mehr Möglichkeiten	221
8.6	F&A	224
9	Dateizugriffe und Ausnahmen	227
9.1	Ein erstes Beispiel für einen Dateizugriff	228
9.1.1	Das Prinzip der Arbeitsteilung	230
9.2	Die Rolle der Streams	232
9.2.1	Die <i>IO.Stream</i> -Klasse	232
9.3	Textdateien schreiben und lesen	233
9.3.1	Die <i>FileStream</i> -Klasse	234
9.3.2	Die <i>StreamReader</i> -Klasse	234
9.3.3	Die <i>StreamWriter</i> -Klasse	234
9.4	Text oder binär – wo liegt der Unterschied?	235
9.5	Überblick über den <i>System.IO</i> -Namensraum	237
9.6	Zugriffe auf das Dateisystem	238
9.7	Arrays in einer Datei speichern und wieder auslesen	240
9.7.1	Sind Sie noch bei mir? – Kurze Pause zum Ausruhen und Reflektieren	241
9.8	Ausnahmen und wie sie abgefangen werden	243
9.8.1	Ausnahmen abfangen	245
9.8.2	Der Try Catch-Befehl	247
9.8.3	Informationen über eine Ausnahme erhalten	247
9.8.4	Die Rolle der <i>SystemException</i> -Klasse	250
9.8.5	Ein allgemeiner Rahmen für eine Ausnahmebehandlung	251
9.8.6	Erweiterung mit dem When-Schlüsselwort	252
9.8.7	Gemeinsamer Ausgang für die Ausnahmebehandlung	252
9.8.8	Ausnahmebehandlung bei verschachtelten Prozeduren	253
9.8.9	Ausnahmen gezielt auslösen	255
9.9	Zusammenfassung	255
9.10	F&A	256

10	Fortgeschrittenere Programmelemente in Visual Basic .NET	259
10.1	Klein und praktisch: der With-Befehl	260
10.2	Der Gültigkeitsbereich von Variablen	261
10.2.1	Variablen in einem Befehlsblock	261
10.2.2	Variablen in einer Funktion (oder Prozedur)	262
10.2.3	Variablen in einer Klasse – privat, öffentlich oder was?	262
10.2.4	Variablen in einem Assembly	264
10.2.5	Nur für Freunde – die Deklaration mit Friends	265
10.3	Typenkonvertierungen	266
10.3.1	Option Strict On	266
10.3.2	Die CType-Funktion	267
10.3.3	Ein Wort zur Konvertierung von Datumsangaben	268
10.3.4	Die »vielen« C<Type>-Funktionen	268
10.3.5	Direkter Zugriff auf einen Typ – die GetType-Methode	269
10.4	Zufallszahlen	270
10.5	Sprünge im Programm – der GoTo-Befehl	271
10.6	Sammeln für einen guten Zweck – die Collection-Klassen	272
10.6.1	Die NameValueCollection-Klasse im Namensraum System.Collections.Specialized	274
10.6.2	Die HashTable-Klasse – ein Klassiker im modernen Gewand	276
10.6.3	Die CollectionBase-Klasse – ohne Vererbung geht es nicht	279
10.7	Spezialist für Auflistungen – die For Each-Schleife	282
10.8	Zugriff auf die Kommandozeilenargumente	284
10.8.1	Spezialität der Konsolenanwendungen	285
10.9	Die Environment-Klasse	285
10.10	Systeminformationen abfragen	286
10.10.1	Der Zugriff auf die Registry	287
10.10.2	Der Aufruf von API-Funktionen	288
10.11	Was sind eigentlich (noch einmal) Namensräume?	290
10.11.1	Der Imports-Befehl	291
10.11.2	Eigene Namensräume definieren	293
10.12	F&A	295
11	OOP für (etwas) Fortgeschrittenes	297
11.1	Die Basisklasse als Vorbild: Vererbung	298
11.1.1	Vererbung Schritt für Schritt kennen lernen	298
11.1.2	Noch einmal das Protected-Schlüsselwort	301
11.1.3	Mehrfache Vererbung kein Problem	301
11.1.4	Die abgeleitete Klasse erweitern	302

11.1.5	Überschreiben von Mitgliedern	303
11.1.6	Die Rolle von MyBase	304
11.1.7	Der Konstruktor der Basisklasse	305
11.1.8	Polymorphie (garantiert ungefährlich)	307
11.1.9	Gründe für Vererbung	309
11.2	Spezialitäten beim Umgang mit Klassen	310
11.2.1	Klassen vor Vererbung schützen	310
11.2.2	Schützen von Mitgliedern	310
11.2.3	Verdecken von Variablen mit Shadows	311
11.2.4	Festlegen einer Standardeigenschaft	312
11.3	Abstrakte Basisklassen	313
11.3.1	Abstrakt oder virtuell?	316
11.4	Völlig unspektakulär: Ereignisse und Delegates	316
11.4.1	Ein Delegate stellt sich vor	316
11.4.2	Der Delegate-Befehl	317
11.4.3	Ein Delegate-Objekt deklarieren	317
11.4.4	Und nun zu den Ereignissen	319
11.4.5	Ein Ereignis über ein Delegate-Objekt definieren	320
11.4.6	Ein Ereignis definieren	320
11.4.7	Ein Ereignis auslösen	320
11.4.8	Eine Ereignisprozedur definieren	321
11.4.9	Der AddHandler-Befehl	322
11.4.10	Ein Wort zu MultiCast-Delegates	322
11.5	Schnittstellen	323
11.5.1	Schnittstellen definieren	324
11.5.2	Schnittstellen in den .NET-Basisklassen	327
11.5.3	Spezialitäten beim Umgang mit Schnittstellen	329
11.5.4	Mehrere Schnittstellen implementieren	329
11.5.5	Feststellen, ob eine Klasse eine Schnittstelle unterstützt	330
11.5.6	Die Vorteile von Schnittstellen	330
11.6	Zusammenfassung	331
11.7	F&A	332
12	XML, was sonst?	335
12.1	XML für »Dummies«	336
12.1.1	Wohl geformte XML-Dokumente	339
12.2	Von Knoten und Kindern	340
12.2.1	Die Rolle der Attribute	342
12.2.2	XML-Dokumente sind in Wirklichkeit Bäume	343
12.2.3	Schemadateien geben XML-Dokumenten eine Struktur	345

12.3	Beispiele mit XML	347
12.3.1	Ein erstes Beispiel mit XML	347
12.3.2	XML-Dateien lesen	349
12.3.3	Der Zugriff auf Attribute	350
12.3.4	Die XMLReader-Klasse	351
12.3.5	Spezialitäten mit XPath	353
12.3.6	XML-Dateien schreiben	355
12.3.7	Objekte speichern durch XML-Serialisierung	358
12.4	Wann ist XML wichtig und wann nicht?	361
12.5	Ein kurzer Ausblick auf die XML-Webservices	362
12.6	Zusammenfassung	364
12.7	F&A	364
13	Visual Basic .NET und das Internet	367
13.1	Internet für »Dummies«	368
13.1.1	Internet = Client + Server	370
13.2	Netzwerkverbindungen über Sockets	372
13.3	Kurzer Überblick über den System.Net-Namensraum	378
13.4	Der Download einer HTML-Datei	379
13.5	Und nun alles mit XML	381
13.6	Zusammenfassung	384
13.7	F&A	384
14	Windows Forms, Controls und Visual Studio .NET	387
14.1	Windows Forms und Controls	388
14.2	Visual Studio .NET stellt sich vor	392
14.2.1	Die IDE im Schnelldurchgang	393
14.2.2	Anlegen von Projekten	399
14.2.3	Projekteigenschaften	400
14.2.4	Bestandteile eines Projekts	401
14.2.5	Dateierweiterungen	402
14.2.6	Debug oder Release?	402
14.3	Windows-Anwendungen	403
14.3.1	Windows Forms und Controls im Programm ansprechen	403
14.3.2	Der Zugriff auf einzelne Windows Forms und Controls	405
14.3.3	Eine Windows-Anwendung Schritt für Schritt	406
14.3.4	Wenn Fehler auftreten – die Aufgabenliste	409
14.4	Zusammenfassung	409
14.5	F&A	410
14.6	Schlusswort und Ausblick	411

A	.NET von A bis Z	413
B	Visual Basic .NET-Referenz	417
B.1	Visual Basic .NET-Befehle	417
B.2	Visual Basic 6.0-Befehle, die nicht mehr unterstützt werden	420
B.3	Visual Basic .NET-Schlüsselwörter	421
B.4	Visual Basic .NET-Operatoren	424
B.5	Die Visual Basic .NET-Datentypen	425
C	Überblick über das .NET-Framework-SDK	427
C.1	Allgemeines zum .NET-Framework-SDK	427
C.2	Die Installation	429
C.3	Die Laufzeitumgebung	430
C.4	Ein Wort zum Editor	431
D	Antworten	433
	Stichwortverzeichnis	473