

Klaus Löffelmann

Visual Basic 2005 – Das Entwicklerbuch

Klaus Löffelmann

Visual Basic 2005 – Das Entwicklerbuch

Microsoft®
Press

Klaus Löffelmann: Visual Basic 2005 – Das Entwicklerbuch
Microsoft Press Deutschland, Konrad-Zuse-Str. 1, 85716 Unterschleißheim
Copyright © 2006 by Microsoft Press Deutschland

Das in diesem Buch enthaltene Programmmaterial ist mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autor, Übersetzer und der Verlag übernehmen folglich keine Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieses Programmmaterials oder Teilen davon entsteht. Die in diesem Buch erwähnten Software- und Hardwarebezeichnungen sind in den meisten Fällen auch eingetragene Marken und unterliegen als solche den gesetzlichen Bestimmungen. Der Verlag richtet sich im Wesentlichen nach den Schreibweisen der Hersteller.

Das Werk, einschließlich aller Teile, ist urheberrechtlich geschützt. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Verlags unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
08 07 06

ISBN 3-86063-537-9

© Microsoft Press Deutschland
(ein Unternehmensbereich der Microsoft Deutschland GmbH)
Konrad-Zuse-Str. 1, D-85716 Unterschleißheim
Alle Rechte vorbehalten

Satz: Silja Brands, Klaus Löffelmann, ActiveDevelop, Lippstadt (<http://ActiveDevelop.de>)
Fachlektorat: Ruprecht Dröge, Ratingen (<http://BeConstructed.de>)
Testing: Jürgen Heckhuis, Lippstadt
Umschlaggestaltung: Hommer Design GmbH, Haar (www.HommerDesign.com)
Layout und Gesamtherstellung: Kösel, Krugzell (www.KoeselBuch.de)

Inhaltsverzeichnis

Am Anfang war	XIX
http://activedevelop.de – Ein wenig Werbung in eigener Sache	XX
Danksagungen	XX
Teil A – Einführung	1
1 Einführung	3
Welche Softwarevoraussetzungen benötigen Sie?	3
Die Visual Basic 2005 Express Edition auf der beiliegenden Buch-CD	4
Wissenswertes zur Installation von Visual Studio 2005	4
Deinstallation von Beta 2, CTP oder Release-Kandidaten	5
Diese Versionen von Visual Studio 2005 gibt es	6
Der Umgang mit Web-Links in diesem Buch – http://links.entwicklerbuch.net	9
Die Begleitdateien zum Buch	10
Nützliches zu Visual Basic 2005 – http://vb2005.de	10
Teil B – Die Visual Studio-Entwicklungsumgebung	11
2 Ein Flug über die Weiten der Visual Studio-IDE	13
Die Startseite – der erste Ausgangspunkt für Ihre Entwicklungen	13
Der Visual Studio-Nachrichten-Channel	14
Anpassen der Liste der zuletzt bearbeiteten Projekte	15
Die IDE auf einen Blick	16
Genereller Umgang mit Fenstern in der Entwicklungsumgebung	18
Wechseln zwischen aktiven Dokumentfenstern mit der Tastatur	20
Die wichtigsten Toolfenster	20
Der Projektmappen-Explorer	21
Das Eigenschaftenfenster	25
Die Fehlerliste	27
Die Aufgabenliste	29
Das Ausgabefenster	30
Die dynamische Hilfe	32
Die Klassenansicht	33
Codeeditor und Designer	34
Die wichtigsten Tastenkombinationen auf einen Blick	34
3 Formular-Designer und Codeeditor enthüllt	37
Das Fallbeispiel – Der DVD-Hüllen-Generator »Covers«	38
Das »Pflichtenheft« von Covers	39
Erstellen eines neuen Projektes	41

Gestalten von Formularen mit dem Windows Forms-Designer	43
Positionieren von Steuerelementen	43
Wem »gehört« eigentlich der Formular-Designer?	45
Häufige Arbeiten an Steuerelementen mit Smarttags erledigen	47
Dynamische Anordnung von Steuerelementen zur Laufzeit	48
Was ist ein Property Extender?	56
Automatisches Scrollen von Steuerelementen in Containern	57
Selektieren von Steuerelementen, die Sie mit der Maus nicht erreichen	60
Festlegen der Tabulatorreihenfolge (Aktivierreihenfolge) von Steuerelementen	60
Über die Eigenschaften Name, Text und Caption	62
Einrichten von Bestätigungs- und Abbrechen-Funktionalitäten für Schaltflächen in Formularen	64
Hinzufügen neuer Formulare zu einem Projekt.	65
Wie geht's weiter?	67
Namensgebungskonventionen für Steuerelemente in diesem Buch	67
Funktionen zum Layouten von Steuerelementen im Designer.	68
Tastaturkürzel für die Platzierung von Steuerelementen	70
Der Codeeditor.	71
Die Wahl der richtigen Schriftart für ermüdungsfreies Arbeiten	71
Viele Wege führen zum Codeeditor	72
IntelliSense – Ihr stärkstes Zugpferd im Coding-Stall	73
Automatische Vervollständigung von Struktur-Schlüsselworten und Codeeintrückung.	75
Fehlererkennung im Codeeditor	76
Smarttags im Editor von Visual Basic.	79
Autokorrektur für intelligentes Kompilieren.	79
Erzwungene Typsicherheit (Option Strict) projektweit einstellen.	80
Erzwungene Typsicherheit für alle folgenden neuen Projekte	80
XML-Dokumentationskommentare für IntelliSense bei eigenen Objekten und Klassen	80
Hinzufügen neuer Codedateien zum Projekt.	84
Code umgestalten (Refactoring)	86
Die Bibliothek der Codeausschnitte (Code Snippets Library)	89
Einstellen des Speicherns von Anwendungseinstellungen mit dem Settings-Designer.	93
Über die Konfigurationseinstellungen »Debug« und »Release« sowie die Geschwindigkeiten der Codeausführung.	100
Weitere Funktionen des Codeeditors	101
Aufbau des Codefensters.	101
Automatischen Zeilenumbruch aktivieren/deaktivieren	102
Navigieren zu vorherigen Bearbeitungspositionen im Code	102
Rechteckige Textmarkierung.	103
Gliederungsansicht	103
Suchen und Ersetzen, Suche in Dateien.	104
Suchen in Dateien	105
Inkrementelles Suchen	106
Gehe zu Zeilennummern.	106
Lesezeichen.	107

4	Tipps & Tricks für das angenehme Entwickeln zuhause und unterwegs	109
	Der Einsatz mehrerer Monitore	109
	Zwei Grafikkarten in einem Rechner?	110
	Und die Bildschirmdarstellung auf Notebooks?	111
	Zurücksetzen der Fenstereinstellungen	112
	Sichern, Wiederherstellen oder Zurücksetzen aller Visual Studio-Einstellungen	112
	Sichern der Visual Studio Einstellungen	112
	Wiederherstellen von Visual Studio-Einstellungen	114
	Zurücksetzen von Visual Studio in den Originalzustand	115
	Wieviel Arbeitsspeicher darf's denn sein?	116
	Testen Ihrer Software unterwegs und zuhause – Microsoft Virtual PC und Microsoft Virtual Server	117
	Microsoft Virtual PC	118
	Virtual Server 2005	120
	Hilfe zur Selbsthilfe	122
	Erweitern Sie die Codeausschnittsbibliothek um eigene Codeausschnitte	127
	Erstellen einer Code Snippets-XML-Vorlage	129
	Hinzufügen einer neuen Snippet-Vorlage zur Snippet-Bibliothek (Codeausschnittsbibliothek)	131
	Verwenden des neuen Codeausschnittes	133
	Parametrisieren von Codeausschnitten	133
Teil C	– Der Umstieg auf Visual Basic 2005	141
5	Der Umstieg von Visual Basic 6.0	143
	Unterschiede in der Variablenbehandlung	144
	Veränderungen bei primitiven Integer-Variablen – die Größen von Integer und Long und der neue Typ Short	144
	Anekdoten aus der Praxis	145
	Typen, die es nicht mehr gibt	145
	... und die primitiven Typen, die es jetzt gibt	146
	Deklaration von Variablen und Variablentypzeichen	148
	Typsicherheit und Typliterale zur Typdefinition von Konstanten	150
	Deklaration und Definition von Variablen »in einem Rutsch«	155
	Vorsicht: New und New können zweierlei in VB6 und VB.NET sein!	155
	Überläufe bei Fließkommazahlen und nicht definierte Zahlenwerte	156
	Alles ist ein Objekt oder »let Set be«	158
	Direktdeklaration von Variablen in For-Schleifen	159
	Unterschiede bei verwendbaren Variablentypen für For/Each in VB6 und VB.NET	161
	Gültigkeitsbereiche von Variablen	161
	Globale bzw. öffentliche (public) Variablen	161
	Variablen mit Gültigkeit auf Modul, Klassen oder Formularebene	162
	Gültigkeitsbereiche von lokalen Variablen	162
	Arrays	164
	Die Operatoren += und -= und ihre Verwandten	165
	Die Bitverschiebeoperatoren << und >>	166
	Fehlerbehandlung	167
	Elegantes Fehlerabfangen mit Try/Catch/Finally	169

Kurzschlussauswertungen mit OrElse und AndAlso	174
Variablen und Argumente auch an Subs in Klammern!	175
Namespaces und Assemblies	176
Assemblies	176
Namespaces	178
So bestimmen Sie Assemblynamen und Namespace für Ihre eigenen Projekte	181
Verschiedene Namespaces in einer Assembly	182
6 Der Umstieg von Visual Basic.NET 2002 und 2003	185
Neue Sprachelemente in Visual Basic 2005	185
Continue in Schleifen	185
Gezieltes Freigeben von Objekten mit Using	186
Zugriff auf den Framework-System-Namespace mit Global	188
Über mehrere Codedateien aufgeteilter Klassencode – Partial Class	190
Übersicht über weitere Neuerungen in Visual Basic 2005	190
Teil D – OOP – Objektorientiertes Programmieren	193
7 Vorüberlegungen zur objektorientierten Programmierung	195
Über Assemblies, Namespaces, CLR, CLI, BCL, JITter und andere .NET-Terminologien. . .	196
Was ist eine Assembly?	196
Was ist ein Namespace?	197
Was versteckt sich hinter CLR (Common Language Runtime) und CLI (Common Language Infrastructure)?	198
Was ist die FCL (Framework Class Library) und was die BCL (die Base Class Library)?	198
Was ist das CTS (Common Type System)?	199
Was ist MS-IML (Microsoft-Intermediate Language) und wozu dient der JITter?.	199
Erzwungene Typsicherheit und Deklarationszwang von Variablen.	201
Namensgebung von Variablen	204
Und welche Sprache ist die beste?	206
Prozedurale Programmierung versus OOP	206
Prozedurale Programmierung ade?	207
8 Auf zum Klassentreffen!	215
Und ab in den Sandkasten	216
Konsolenanwendung in VB.NET	216
Das Klassenprinzip am einfachsten Beispiel	218
Das Klassenprinzip am eigenen Beispiel	220
Statische und nicht-statische Methoden und Variablen	221
Nicht verwirren lassen: Static und Shared in VB	223
Smarttags im Editor von Visual Basic	225
Kleiner Exkurs – womit startet ein Programm?	227
Mit Sub New bestimmen, was beim Instanzieren passiert – der Klassenkonstruktor	227
Überflüssige Funktionen mit dem Obsolete-Attribut markieren.	230
Überladen von Funktionen und Konstruktoren	231
Methodenüberladung und optionale Parameter	233
Gegenseitiges Aufrufen von überladenen Methoden	234

Gegenseitiges Aufrufen von überladenen Konstruktoren	236
Hat jede Klasse einen Konstruktor?	237
Zusätzliche Werkzeuge für .NET	238
Statische Konstruktoren und Variablen	241
Eigenschaften	246
Zuweisen von Eigenschaften	247
Ermitteln von Eigenschaften	247
Nur-Lesen und Nur-Schreiben-Eigenschaften.	249
Eigenschaften mit Parametern.	250
Überladen von Eigenschaften	251
Statische Eigenschaften	252
Default-Eigenschaften (Standardeigenschaften)	252
Öffentliche Variablen oder Eigenschaften – eine Glaubensfrage?	255
Zugriffsmodifizierer von Klassen, Prozeduren, Eigenschaften und Variablen.	256
Zugriffsmodifizierer bei Klassen	257
Zugriffsmodifizierer bei Prozeduren (Subs, Functions, Properties)	257
Zugriffsmodifizierer bei Variablen	258
Unterschiedliche Zugriffsmodifizierer für Eigenschaften-Accessors	259
9 Klassenvererbung und Polymorphie	261
Wiederverwendbarkeit von Klassen durch Vererbung (Inheritance)	261
Initialisierung von Member-Variablen bei Klassen ohne Standardkonstruktoren	270
Überschreiben von Methoden und Eigenschaften.	271
Überschreiben vorhandener Methoden und Eigenschaften von Framework-Klassen	274
Das Speichern von Objekten im Arbeitsspeicher – und die daraus resultierende Vorsicht mit ihnen	275
Polymorphie.	279
Zahlen mit ToString formatiert in Zeichenketten umwandeln	287
Polymorphie und der Gebrauch von Me, MyClass und MyBase	293
Abstrakte Klassen und virtuelle Prozeduren	294
Eine Klasse mit MustInherit als abstrakt deklarieren.	295
Eine Methode oder Eigenschaft einer abstrakten Klasse mit MustOverride als virtuell deklarieren.	296
Schnittstellen (Interfaces)	297
Unterstützung bei abstrakten Klassen und Schnittstellen durch den Editor.	304
Schnittstellen, die Schnittstellen implementieren	309
Einbinden mehrere Schnittstellen in eine Klasse.	310
Die Methoden und Eigenschaften von Object.	311
Polymorphie am Beispiel von ToString und der ListBox	312
Prüfen auf Gleichheit von Objekten mit Object.Equals oder dem Is/IsNot-Operator	315
Equals, Is und IsNot im praktischen Entwicklungseinsatz.	317
Übersicht über die Eigenschaften und Methoden von Object	318
Shadowing (Überschatten) von Klassenprozeduren	318
Shadows als Unterbrecher der Klassenhierarchie	320
Sonderform »Modul« in Visual Basic	324
Singleton-Klassen und Klassen, die sich selbst instanzieren	324

10	Über Structure und den Unterschied zwischen Referenz- und Werttypen	327
	Der Unterschied zwischen Referenz- und Werttyp	327
	Erstellen von Werttypen mit Structure am praktischen Beispiel	329
	Unterschiedliche Verhaltensweisen von Werte- und Referenztypen	334
	Verhalten der Parameterübergabe mit ByVal und ByRef steuern	336
	Konstruktoren und Standardinstanzierungen von Werttypen	336
	Gezieltes Zuweisen von Speicherbereichen für Struktur-Member mit den StructLayout- und FieldOffset-Attributen	338
	Performance-Unterschiede zwischen Werte- und Referenztypen	341
	Wieso kann durch Vererbung aus einem Object-Referenztyp ein Werttyp werden?	342
11	Typumwandlungen (Type Casting) und Boxing von Datentypen	343
	Konvertieren von primitiven Typen	344
	Konvertieren von und in Zeichenketten (Strings)	345
	Konvertieren von Strings mit den Parse- und ParseExact-Methoden	346
	Konvertieren in Strings mit der ToString-Methode	346
	Abfangen von fehlschlagenden Typkonvertierungen mit TryParse oder Ausnahmebehandlern	347
	Casten von Referenztypen mit DirectCast	348
	Boxing von Werttypen und primitiven Typen	349
	Zufallszahlen mit der Random-Klasse	350
	Was DirectCast nicht kann	352
	Boxen beim Implementieren von Schnittstellen in Strukturen	352
12	Beerdigen von Objekten – Dispose, Finalize und der Garbage Collector	355
	Der Garbage Collector – die Müllabfuhr in .NET	357
	Generationen	358
	Finalize	359
	Wann Finalize nicht stattfindet	361
	Dispose	364
	Unterstützung durch den Visual Basic-Editor beim Einfügen eines Disposable- Patterns	374
13	Operatoren für benutzerdefinierte Typen	377
	Einführung in Operatorenprozeduren	378
	Vorbereitung einer Struktur oder Klasse für Operatorenprozeduren	379
	Implementierung von Rechenoperatoren	383
	Überladen von Operatorenprozeduren	384
	Implementierung von Vergleichsoperatoren	385
	Implementierung von Typkonvertierungsoperatoren mit Operator CType	386
	Implementieren von Wahr- und Falsch-Auswertungsoperatoren	387
	Problembehandlungen bei Operatorenprozeduren	389
	Aufgepasst bei der Verwendung von Referenztypen	389
	Mehrdeutigkeiten bei der Auflösung von Signaturen	390
	Übersicht der implementierbaren Operatoren	391
14	Generische Klassen und Strukturen (Generics)	393
	Verwenden einer Codebasis für verschiedene Typen	393

Lösungsansätze	394
Typengeneralisierung durch den Einsatz generischer Datentypen	397
Beschränkungen (Constraints)	400
Beschränkungen für generische Typen auf eine bestimmte Basisklasse	400
Beschränkungen auf Klassen, die bestimmte Schnittstellen implementieren	405
Beschränkungen auf Klassen, die über einen Standardkonstruktor verfügen	408
Beschränkungen auf Wertetypen	409
Kombinieren von Beschränkungen und Bestimmen mehrerer Typparameter	409
Vererben von generischen Typen	410
15 Ereignisse und Delegates	413
Konsumieren von Ereignissen mit WithEvents und Handles	415
Auslösen von Ereignissen	418
Der Umweg über Onxxx	419
Zur-Verfügung-Stellen von Ereignisparametern	420
Die Quelle des Ereignisses: Sender	420
Nähere Informationen zum Ereignis: EventArgs	422
Dynamisches Anbinden von Ereignissen mit AddHandler	423
Delegates	432
Teil E – Die wichtigsten Datentypen des Frameworks	437
16 Primitive Datentypen	439
Einführung	440
.NET-Äquivalente primitiver Datentypen	440
Numerische Datentypen	442
Numerische Datentypen deklarieren und definieren	442
Delegation numerischer Berechnungen an den Prozessor	442
Hinweis zur CLS-Konformität	444
Die numerischen Datentypen auf einen Blick	445
Rundungsfehler bei der Verwendung von Single und Double	451
Besondere Funktionen, die für alle numerischen Datentypen gelten	454
Spezielle Funktionen der Fließkommatypen	457
Spezielle Funktionen des Wertetyps Decimal	459
Der Datentyp Char	459
Der Datentyp String	460
Strings – gestern und heute	461
Strings deklarieren und definieren	461
Der String-Konstruktor als Ersatz von String\$	462
Einem String Zeichenketten mit Sonderzeichen zuweisen	463
Speicherbedarf von Strings	464
Strings sind unveränderlich	464
Speicheroptimierung von Strings durch das Framework	464
Ermitteln der String-Länge	465
Ermitteln von Teilen eines Strings oder eines einzelnen Zeichens	466
Angleichen von String-Längen	466
Suchen und Ersetzen	467
Algorithmisches Auflösen eines Strings in Teile	470

Ein String-Schmankerl zum Schluss	472
Iterieren durch einen String	475
StringBuilder vs. String – wenn es auf Geschwindigkeit ankommt	476
Wrapper-Klassen für Betriebssystemaufrufe	480
Der Datentyp Boolean	482
Konvertieren von und in numerische Datentypen.	483
Konvertierung von und in Strings	483
Vergleichsoperatoren, die boolesche Ergebnisse zurückliefern	484
Anweisungen, die bedingten Programmcode ausführen	485
Der Datentyp Date	488
Rechnen mit Zeiten und Datumswerten – TimeSpan	489
Bibliothek mit brauchbaren Datumsrechenfunktionen	490
Zeichenketten in Datumswerte wandeln	493
17 Kulturabhängiges Formatieren von Zahlen- und Datumswerten	497
Allgemeines über Format Provider in .NET	497
Kulturabhängige Formatierungen mit CultureInfo.	498
Vermeiden von kulturabhängigen Programmfehlern	500
Formatierung durch Formatzeichenfolgen	501
Formatierung von numerischen Ausdrücken durch Formatzeichenfolgen	501
Formatierung von numerischen Ausdrücken durch vereinfachte Formatzeichenfolgen.	504
Formatierung von Datums- und Zeitausdrücken durch Formatzeichenfolgen	505
Formatierung von Zeitausdrücken durch vereinfachte Formatzeichenfolgen	510
Gezielte Formatierungen mit Format Providern.	511
Gezielte Formatierungen von Zahlenwerten mit NumberFormatInfo.	511
Gezielte Formatierungen von Zeitwerten mit DateTimeFormatInfo	512
Kombinierte Formatierungen.	513
Ausrichtungen in kombinierten Formatierungen	514
Angaben von Formatzeichenfolgen in den Indexkomponenten.	516
So helfen Ihnen benutzerdefinierte Format Provider, Ihre Programme zu internationalisieren	517
Benutzerdefinierte Format Provider durch IFormatProvider und ICustomFormatter.	527
Automatisch formatierbare Objekte durch Einbinden von IFormattable.	530
18 Enums (Aufzählungen)	533
Bestimmung der Werte der Aufzählungselemente	534
Bestimmung der Typen der Aufzählungselemente	535
Ermitteln des Elementtyps zur Laufzeit.	535
Konvertieren von Enums	536
In Zahlenwerte umwandeln und aus Werten definieren	536
In Strings umwandeln und aus Strings definieren	536
Flags-Enum (Flags-Aufzählungen)	537
Abfrage von Flags-Aufzählungen	538
19 Arrays und Auflistungen (Collections)	539
Grundsätzliches zu Arrays	540
Arrays als Parameter und Funktionsergebnis	541

Änderung der Array-Dimensionen zur Laufzeit	542
Wertvorbelegung von Array-Elementen im Code	547
Mehrdimensionale und verschachtelte Arrays.	547
Die wichtigsten Eigenschaften und Methoden von Arrays	548
Implementierung von Sort und BinarySearch für eigene Klassen	551
Enumeratoren	558
Benutzerdefinierte Enumeratoren mit IEnumerable	559
Grundsätzliches zu Auflistungen	562
Die wichtigen Auflistungen der Base Class Library.	565
ArrayList – universelle Ablage für Objekte	565
Typsichere Auflistungen auf Basis von CollectionBase	568
Hashtables – für das Nachschlagen von Objekten.	571
Anwenden von Hashtables	572
Verwenden eigener Klassen als Schlüssel (Key).	581
Enumerieren von Datenelementen in einer Hashtable	585
Typsichere Hashtable	585
Queue – Warteschlangen im FIFO-Prinzip.	587
Stack – Stapelverarbeitung im LIFO-Prinzip.	589
SortedList – Elemente ständig sortiert halten	590
Sortierung einer SortedList nach beliebigen Datenkriterien	591
20 Arbeiten mit generischen Typen und generischen Auflistungen	595
Wertetypen, die Nothing speichern können – Nullable(Of)	595
Besonderheiten bei Nullable(Of) beim Boxen.	597
Generische Auflistungen (Generic Collections)	599
KeyedCollection – Schlüssel/Wörterbuch-Auflistung mit zusätzlichem Index- Abrufen.	602
Elementverkettungen mit LinkedList(Of)	605
Auflistungen und Aktionen (Actions), Aussageprüfer (Predicates) und Vergleiche (Comparisons)	607
ForEach und die generische Action-Klasse	608
Sort und die generische Comparison-Klasse	609
Find und die generische Predicate-Klasse	610
21 Reguläre Ausdrücke (Regular Expressions)	613
RegExperimente mit dem RegExplorer	614
Erste Gehversuche mit Regulären Ausdrücken	616
Einfache Suchvorgänge	616
Einfache Suche nach Sonderzeichen	617
Komplexere Suche mit speziellen Steuerzeichen	618
Verwendung von Quantifizierern	620
Gruppen	622
Suchen und Ersetzen	624
Captures	625
Optionen bei der Suche	627
Steuerzeichen zu Gruppendefinitionen	629
Programmieren von Regulären Ausdrücken.	630
Ergebnisse im Match-Objekt.	630

Die Matches-Auflistung	631
Abrufen von Captures und Gruppen eines Match-Objektes	632
Regex am Beispiel: Berechnen beliebiger mathematischer Ausdrücke.	634
Der Formelparser	635
Die Klasse ADFormularParser	636
Vererben der Klasse ADFormularParser, um eigene Funktionen hinzuzufügen	654
22 Serialisierung von Objekten	657
Einführung in Serialisierungstechniken	658
Serialisieren mit SoapFormatter und BinaryFormatter	660
Flaches und tiefes Klonen von Objekten.	666
Universelle DeepClone-Methode	669
Serialisieren von Objekten mit Zirkelverweisen	671
Serialisierung von Objekten unterschiedlicher Versionen beim Einsatz von BinaryFormatter oder SoapFormatter-Klassen	674
XML-Serialisierung	674
Prüfen der Versionsunabhängigkeit der XML-Datei	679
Serialisierungsfehler bei KeyedCollection	680
Workaround	684
23 Attribute und Reflection.	687
Genereller Umgang mit Attributen.	688
Einsatz von Attributen am Beispiel von ObsoleteAttribute	688
Die speziell in Visual Basic verwendeten Attribute	690
Einführung in Reflection	691
Die Type-Klasse als Ausgangspunkt für alle Typenuntersuchungen	692
Klassenanalysefunktionen, die ein Type-Objekt bereitstellt.	694
Objekthierarchie von MemberInfo und Casten in den spezifischen Info-Typ	696
Ermitteln von Eigenschaftswerten über PropertyInfo zur Laufzeit	696
Erstellung benutzerdefinierter Attribute und deren Erkennen zur Laufzeit	697
Ermitteln von benutzerdefinierten Attributen zur Laufzeit	700
Teil F – Vereinfachungen in Visual Basic 2005.	703
24 Eine philosophische Betrachtung der Vereinfachungen in Visual Basic 2005.	705
Die VB2005-Vereinfachungen am Beispiel DotNetCopy	706
DotNetCopy mit /Autostart und /Silent-Optionen	709
Exkurs: Die Klassen FileInfo- und DirectoryInfo zur Pflege von Verzeichnissen und Dateien	709
Die prinzipielle Funktionsweise von DotNetCopy	711
25 Der My-Namespace	713
Formulare ohne Instanzierung aufrufen.	715
Auslesen der Befehlszeilenargumente mit My.Application.CommandLineArgs	716
Gezieltes Zugreifen auf Ressourcen mit My.Resources	718
Anlegen und Verwalten von Ressource-Elementen.	718
Abrufen von Ressourcen mit My.Resources.	719

Internationalisieren von Anwendungen mithilfe von Ressource-Dateien und dem My-Namespace.	721
Vereinfachtes Durchführen von Dateioperationen mit My.Computer.FileSystem	724
Verwenden von Anwendungseinstellungen mit My.Settings	727
26 Das Anwendungsframework	731
Die Optionen des Anwendungsframeworks	733
Windows XP Look and Feel für eigene Windows-Anwendungen – Visuelle XP-Stile aktivieren	733
Verhindern, dass Ihre Anwendung mehrfach gestartet wird – Einzelinstanzanwendung erstellen.	733
MySettings-Einstellungen automatisch sichern – Eigene Einstellungen beim Herunterfahren speichern.	733
Bestimmen, welcher Benutzer-Authentifizierungsmodus verwendet wird.	733
Festlegen, wann eine Anwendung »zu Ende« ist – Modus für das Herunterfahren	734
Einen Splash-Dialog beim Starten von komplexen Anwendungen anzeigen – Begrüßungsdialog	734
Eine Codedatei implementieren, die Anwendungsereignisse (Starten, Beenden, Netzwerkzustand, generelle Fehler) behandelt	734
Teil G – Entwickeln von SmartClient-Anwendungen	739
27 Programmieren mit Windows Forms	741
Strukturieren von Daten in Windows Forms-Anwendungen	743
Formulare zur Abfrage von Daten verwenden – Aufruf von Formularen aus Formularen	754
Der Umgang mit modalen Formularen	755
Überprüfung auf richtige Benutzereingaben in Formularen	766
Anekdoten aus der Praxis	767
Anwendungen über die Tastatur bedienbar machen.	771
Definition von Schnellzugriffstasten per »&<-Zeichen	771
Accept- und Cancel-Schaltflächen in Formularen.	774
Über das »richtige« Schließen von Formularen.	775
Unsichtbarmachen eines Formulars mit Hide	775
Schließen des Formulars mit Close	775
Entsorgen des Formulars mit Dispose.	776
Grundsätzliches zum Darstellen von Daten aus Auflistungsklassen in Steuerelementen.	777
Darstellen von Daten aus Auflistungen im ListView-Steuerelement	777
Spaltenköpfe	778
Hinzufügen von Daten	778
Beschleunigen des Hinzufügens von Elementen	779
Automatisches Anpassen der Spaltenbreiten nach dem Hinzufügen aller Elemente	779
Zuordnen von ListViewItem und Objekten, die sie repräsentieren	780
Feststellen, dass ein Element der Liste selektiert wurde.	781
Selektieren von Elementen in einem ListView-Steuerelement.	782
Verwalten von Daten aus Auflistungen mit dem DataGridView-Steuerelement	782
Über Datenbindung, Bindungsquellen, die DataSource-Komponente und warum Currency nicht unbedingt Währung heißt	784
Erstellen einer gebundenen DataGridView mit dem Visual Basic-Designer	786

Sortieren und Benennen der Spalten eines DataGridView-Steuerelements	791
Programmtechnisches Abrufen der aktuellen Zeile und aktuellen Spalte.	796
Formatieren von Zellen.	797
Problemlösung für das Parsen eines Zellenwerts mit einem komplexen Anzeigeformat.	797
Verhindern des Editierens bestimmter Spalten aber Gestatten ihrer Neueingabe	799
Überprüfen der Richtigkeit von Eingaben auf Zellen- und Zeilenebene	800
Ändern des Standardpositionierungsverhaltens des Zellencursors nach dem Editieren einer Zelle	803
Entwickeln von MDI-Anwendungen	805
Vererben von Formularen	813
ControlCollection vs. Steuerelemente-Array aus VB6	814
Ein erster Blick auf den Designer-Code eines Formulars.	817
Modifizierer von Steuerelementen in geerbten Formularen.	820
28 Im Motorraum von Formularen und Steuerelementen.	821
Über das Vererben von Form und die Geheimnisse des Designer-Codes	821
Geburt und Tod eines Formulars – New und Dispose.	823
Ereignisbehandlung von Formularen und Steuerelementen	827
Vom Programmstart mithilfe des Anwendungsframeworks über eine Benutzeraktion zur Ereignisauslösung	828
Implementieren neuer Steuerelementereignisse auf Basis von Warteschlangenauswertungen.	833
Wer oder was löst welche Formular- bzw. Steuerelementereignisse wann aus?.	834
Kategorie Erstellen und Zerstören des Formulars	836
Kategorie Mausereignisse des Formulars.	839
Kategorie Tastaturereignisse des Formulars	841
Kategorie Position und Größe des Formulars	842
Kategorie Anordnen der Komponenten und Neuzeichnen des Formulars	843
Kategorie Fokussierung des Formulars	845
Kategorie Tastaturvorverarbeitungsnachrichten des Formulars	846
Kategorie Erstellen/Zerstören des Controls (des Steuerelements)	847
Kategorie Mausereignisse des Controls.	848
Kategorie Tastaturereignisse des Controls.	850
Kategorie Größe und Position des Controls	851
Kategorie Neuzeichnen des Controls und Anordnen untergeordneter Komponenten.	853
Kategorie Tastaturnachrichtenvorverarbeitung des Controls	855
Die Steuerungsroutinen des Beispielprogramms	856
Anmerkungen zum Beispielprogramm	861
29 GDI+ zum Zeichnen von Formular- und Steuerelementinhalten verwenden	863
Einführung in GDI+.	864
Linien, Flächen, Pens und Brushes	867
Angabe von Koordinaten	868
Wieso Integer- und Fließkommaangaben für Positionen und Ausmaße?	869
Wie viel Platz habe ich zum Zeichnen?	869
Das gute, alte Testbild und GDI+ im Einsatz sehen!	869
Exaktes Einpassen von Text mit GDI+-Skalierungsfunktionen	876

Flimmerfreie, fehlerfreie und schnelle Darstellungen von GDI+-Zeichnungen.	878
Zeichnen ohne Flimmern	879
Programmtechnisches Bestimmen der Formulargröße	882
Was Sie beim Zeichnen von breiten Linienzügen beachten sollten	884
30 Entwickeln von Steuerelementen	893
Neue Steuerelemente auf Basis vorhandener Steuerelemente implementieren	894
Der Weg eines Steuerelements vom Klassencode in die Toolbox.	896
Implementierung von Funktionslogik und Eigenschaften	898
Steuern von Eigenschaften im Eigenschaftfenster	900
Entwickeln von konstituierenden Steuerelementen	903
Anlegen eines Projekts für die Erstellung einer eigenen Steuerelement-Assembly.	904
Initialisieren des Steuerelements	908
Methoden und Ereignisse delegieren.	911
Implementieren der Funktionslogik	912
Implementierung der Eigenschaften	914
Erstellen von Steuerelementen von Grund auf	916
Ein Label, das endlich alles kann	916
Vorüberlegungen und Grundlagenerarbeitung	917
Klasseninitialisierungen und Einrichten der Windows-Darstellungsstile des Steuerelements	918
Zeichnen des Steuerelements	920
Der Unterschied zwischen Refresh, Invalidate und Update.	923
Größenbeeinflussung durch andere Eigenschaften	924
Implementierung der Blink-Funktionalität	926
Designercode-Generierung und Zurücksetzen von werteerbenden Eigenschaften mit ShouldSerializeXXX und ResetXXX	929
Designer-Reglementierungen	931
31 Mehreres zur gleichen Zeit erledigen – Threading in .NET	933
Threads durch ein Thread-Objekt initiieren.	936
Starten von Threads	938
Grundsätzliches über Threads.	939
Synchronisieren von Threads	939
Synchronisieren der Codeausführung mit SyncLock.	939
Mehr Flexibilität in kritischen Abschnitten mit der Monitor-Klasse	943
Synchronisieren von beschränkten Ressourcen mit Mutex	946
Weitere Synchronisierungsmechanismen	949
Verwenden von Steuerelementen in Threads	953
Managen von Threads	954
Starten eines Threads mit Start	954
Vorübergehendes Aussetzen eines Threads mit Sleep – Statusänderungen im Framework bei Suspend und Resume	954
Abbrechen und Beenden eines Threads.	955
Datenaustausch zwischen Threads durch Kapseln von Threads in Klassen.	959
Der Einsatz von Thread-Klassen in der Praxis.	962
Verwenden des Thread-Pools.	970
Thread-sichere Formulare in Klassen kapseln	975

Threads durch den Background-Worker initiieren.	978
Threads durch asynchrone Aufrufe von Delegates initiieren	981
32 SQL Server 2005 und ADO.NET.	983
SQL Server 2005 Express	984
Einsatz von SQL Server Express in eigenen Anwendungen	985
Installation von SQL Server 2005 Express unter Windows XP im »gemischten Modus«	986
Konfiguration der Netzwerkeinstellungen und Einrichten der Firewall unter Windows XP SP2	991
Grundsätzliches zu Datenbanken	992
Aufbau der Beispieldatenbank.	992
Klärung grundsätzlicher Begriffe.	994
Einsehen von Daten mit dem Server-Explorer.	995
Anekdoten aus der Praxis	997
Programmieren mit ADO.NET	998
Verbindungen zur Datenbank mit der Connection-Klasse herstellen und Befehle mit der Command-Klasse ausführen.	998
Datenverbindungen herstellen und Resultsets mit der DataReader-Klasse auslesen . . .	999
Unverbundene Daten mit dem DataTable-Objekt verwalten.	1001
Einige SELECT-Beispiele:	1006
Ändern und Ergänzen von Daten in Datentabellen.	1009
Was machte das »alte« ADO?	1011
Verwenden von DataAdapter, DataTable und Command-Objekten zur Übermittlung von Aktualisierungen	1012
Für »Mal-Eben-Abfragen« – die CommandBuilder-Klasse	1019
DataSets und typisierte DataSets.	1021
Erstellen eines typisierten DataSets mit der Visual Studio IDE	1022
Grundsätzliches zu typisierten DataSets	1025
Erweitern des TableAdapters um zusätzliche Abfragefunktionen, denen Parameter übergeben werden können.	1027
Erstellen von datenbankgebundenen Formularen in Windows Forms-Anwendungen .	1033
Und so geht es weiter	1037
Stichwortverzeichnis	1039

Am Anfang war ...

... eine erste Beta-Version, wie immer keine Zeit, aber dennoch nach dem ersten Blick soviel Ablenkung durch Begeisterung, dass andere Termine fast platzten. Warum? Ich sah mich mit der ersten »stabilen« Beta-Version von Visual Studio 2005 konfrontiert und war davon so aus dem Häuschen, dass ich mich mehrere Tage damit beschäftigte. Das ist jetzt schon gut eineinhalb Jahre her, aber ich kann mich noch sehr genau daran erinnern, dass ich mit einem solchen Enthusiasmus, der bei mir damals aufkam, nach dem Umstieg von Visual Basic 6 auf Visual Basic .NET nicht mehr gerechnet hatte. Denn schon damals, also vor ca. 3 Jahren, war ich ebenfalls enorm begeistert – die Leser, die das Vorwort des Vorläufers dieses Buchs gelesen haben, wissen warum – und ich glaubte nicht, dass man auf ein schon geniales Entwicklungssystem, wie es Visual Studio .Net 2003 schnell für mich wurde, noch eins 'draufsetzen konnte. Ich wurde eines Besseren belehrt.

Diese Begeisterung hatte Konsequenzen, denn auch wenn es im Juli 2004 noch so aussah, dass die fertige Version von Visual Studio 2005 noch in weiter Ferne lag, traf ich eine mehr emotionale als rationale Entscheidung: Ich entschloss mich, ein Projekt mit einem Umfang von über 1000 Mannstunden bereits mit dieser Version zu beginnen. Und um es gleich vorweg zu nehmen: Durch das frühe Stadium von Visual Studio 2005 hatte ich Erlebnisse, die mich mehr als einmal daran zweifeln ließen, ob ich die richtige Entscheidung getroffen hatte. Bestimmte Sachen, die noch nicht richtig funktionierten, schmissen unser Team teils Wochen zurück. Andere Sachen, die funktionierten, wurden aus der finalen Version aus diesen oder jenen Gründen verbannt, und wir mussten ganze Funktionsblöcke unserer Software neu konzipieren und natürlich auch neu entwickeln. Aber glauben Sie mir: Letzten Endes bereue ich die Entscheidung auf keinen Fall. Mit dem Erscheinen von Visual Studio 2005 ist auch unser Projekt abgeschlossen. Und unsere Anwendung sieht dank Visual Studio 2005 so aus, wie moderne Anwendungsprogramme heutzutage aussehen sollten. Es verfügt über eine Benutzeroberfläche, die mit Visual Studio .NET nie in der gleichen Zeit machbar gewesen wäre. Es lief selbst in der ersten Version unglaublich stabil – unsere Tester hatten nie weniger Arbeit bei den ersten Performance- und Zuverlässigkeitstests – und die Software machte selbst bei den ersten Testinstallationen für den Produktiveinsatz bemerkenswert wenige Probleme. Ohne uns selbst loben zu wollen, liegt das sicherlich auch daran, dass ein paar wirklich erfahrene .NET-Entwickler am Werk waren. Doch ist das auch zu einem abermals so großen Teil der darunter liegenden Plattform – in diesem Fall SQL Server 2005 und .NET-Framework 2.0 – zu verdanken; eine Tatsache übrigens, die alle Beteiligten unseres Entwicklungsteams angenehm überraschte, denn schließlich arbeiteten wir bis zu letzt mit nicht vollständig getesteten Beta-Versionen von Visual Studio 2005.

Was diese 2005er Version von Visual Studio bzw. Visual Basic angeht: Die Jungs aus Redmond haben sich wirklich was einfallen lassen und nicht nur tonnenweise Gehirnschmalz in neue, sinnvolle Funktionen investiert, sondern auch dafür gesorgt, dass Anwendungen, die auf diesem System aufbauen, zuverlässiger agieren als je zuvor.

So ganz nebenbei haben wir uns in der Zeit natürlich auch noch eine ganze Menge an Know-how angeeignet, das inzwischen absolut praxiserprobt ist und Ihnen in diesem Buch zu Gute kommt – so meine ich zu mindest.

Ich hoffe in diesem Sinne, Ihnen mit diesem Buch viele der Erfahrungen weiter geben zu können, die wir im Laufe der letzten Monate gesammelt haben, und dieses Buch dadurch zu einem richtigen Entwicklerbuch zu machen, das nicht nur bloße Theorie transportiert, sondern Ihnen wirklich brauchbare Tipps für die tägliche Praxis liefern kann.

Auf Ihre Reaktionen, die Sie mir an entwicklerbuch@activedevelop.de senden können, bin ich wirklich gespannt!

Klaus Löffelmann, Lippstadt im März 2006.

<http://activedevelop.de> – Ein wenig Werbung in eigener Sache

Man bekommt nicht oft die Möglichkeit, für sein eigenes Gewerbe ohne größeren finanziellen Aufwand werben zu können. Umso mehr mag ich natürlich die Gelegenheit am Schopf packen, um an dieser Stelle ein wenig Öffentlichkeitsarbeit für mein eigenes Dienstleistungsunternehmen zu betreiben. So denn: Falls Sie Bedarf an

- Beratungsdienstleistungen in Sachen .NET 1.1, 2.0 und SQL Server 2005,
- Schulungen (auch inhouse) mit den Schwerpunkten »Umstieg von VB6«, SQL Server 2005 und ADO.NET 2.0, C# 2005 und Visual Basic 2005,
- Dokumentationserstellung, Übersetzungen und Softwarelokalisierungen oder
- Software Development Outsourcing

haben, informieren Sie sich einfach auf unserer Website unter <http://activedevelop.de>, und setzen Sie sich mit uns in Verbindung.

Danksagungen

Das Schreiben dieses Buches war mal wieder ein hartes Stück an Arbeit und wäre ohne entsprechende Unterstützung von Freunden, Familie und den Mitarbeitern von Microsoft und Microsoft Press nicht möglich gewesen. Deswegen möchte ich zunächst **Thomas Braun-Wiesholler**, meinem Lektor, für die Unterstützung in Form von Software, Betriebssystemen, Links, Büchern und vor allen Dingen unzähligen Betaversionen von Visual Studio 2005 bedanken. Unzähligen? Doch nicht: Ich hab sie alle auf einer Spindel gesammelt, und, lieber Thomas, du hast seit der Beta 1 von Visual Studio 2005 im April 2004 nicht weniger als 56 DVDs mit den verschiedensten Versionen von Visual Studio, dem Foundation-Server, der MSDN und Yukon mit heldenhaftem Einsatz organisiert, gebrannt und mir meistens mit der teureren Samstagzustellung geschickt. Und außerdem, das sei hier in aller Deutlichkeit gesagt: Mir macht es echt Spaß, mit dir zusammenzuarbeiten!

Ein ganz dickes Dankeschön an dieser Stelle auch dem Profi aller Profis in Sachen SQL Server **Ruprecht Dröge**, der das Fachlektorat dieses Buches übernommen hat, und mit dem es richtig, richtig lustig ist, zusammenzuarbeiten.

Um die Überprüfung der Lauffähigkeit aller Beispielprogramme und die Kontrolle der richtigen Pfadangaben im Buch hat sich darüber hinaus mein Bürogemeinschaftskollege und guter Freund **Jürgen Heckhuis** gekümmert – auch ihm sei dafür ausdrücklich gedankt!

Dank gilt auch meinen Eltern Gabi und Arnold sowie meinen Freunden, die mich wie beim letzten Buch durch regelmäßiges Besuchen und mich ab und zu entführen gezeigt haben, dass es auch noch »eine Welt da draußen« gibt. Dazu zählen insbesondere Claudia, Christian, Gaby, Miri, Momo und Uta. Und natürlich Uwe Thiemann und mein irischer Freund Gareth Clarke: *Go n'éirí leat, go n' éirí an bóthar leat is céad míle beannachta. Sin sin, níl aon scéal eile agam.*

Und lieber Hartmut Woerrlein. Du bist ein guter Freund und darüber hinaus der Mensch, den ich in dieser verrückten Branche, in der wir arbeiten, mit über 18 Jahren schon am längsten kenne. Und Du hast in Deinem ComputerBILD-Lexikon einen Vorwortkrieg angezettelt, auf den ich mich nur zu gerne einlasse. Du willst es so. Ich habe Dich in meinem letzten Buch also nicht erwähnt und Dir schon gar nichts gewidmet? Wie konnte ich, ich Rüpel!? Das will ich hiermit wieder gut machen. Ich widme Dir deswegen als kleine Stichelei das Stichwortverzeichnis dieses Buches! ;-)

Eine kleine Anekdote in Sachen Motivation

Vor ca. 20 Jahren habe ich eine gute Freundin kennen gelernt. Diese Freundin habe ich bis vor 7 Jahren als einen Menschen gekannt, die mit ihrem Leben scheinbar zufrieden war. Als Zahnarzthelferin hat sie ihren beruflichen Werdegang begonnen, und keiner aus unserem Freundeskreis hat es jemals auch nur in Erwägung gezogen, dass sie sich in Sachen Job und Karriere auch nur irgendwann einmal in ihrem Leben irgendetwas anderes vorstellen würde.

Zwischendurch haben wir uns für einen gewissen Zeitraum aus den Augen verloren. Und so freute ich mich sehr, als ich sie nach einiger Zeit zufällig wieder traf. Doch was folgte, verschlug mir die Sprache: Sie erzählte mir von Software Requirements, Pflichtenheften und Projektplanungen. Sie verwickelte mich in ein Gespräch über objektorientiertes Programmieren, Klassen, Java, .NET und Entwicklungsumgebungen – doch das Gespräch war vergleichsweise einseitig. Sie beobachtete mich mit einem triumphierenden aber ganz bestimmt verdienten Grinsen auf dem Gesicht, wie ich vor ungläubigem Staunen eigentlich nur stillschweigend ihren Worten zu folgen versuchte.

Es stellte sich heraus: Sie hatte hart gearbeitet und tatsächlich eine Ausbildung hingelegt, für die sie sich jeden Tag aufs Neue motivieren müssen. Und leicht fiel ihr das beileibe nicht. Aber sie hat sich durchgebissen, es geschafft und schließlich einen nicht schlecht dotierten Job in »ihrer« neuen Branche gefunden, der ihr Spaß macht, und in dem sie endlich zu zeigen in der Lage war und ist, was sie wirklich kann.

Sich zu motivieren und sich täglich wieder den Herausforderungen zu stellen, gehört gerade in der Branche, in der wir uns bewegen, zu den Eigenschaften, die enorm wichtig sind. Mindestens genau so wichtig, wie das Beherrschen des Handwerkzeugs. Mich hat das, was sie geleistet und erreicht hat, enorm beeindruckt, und ich finde es deswegen auch so erwähnenswert.

Diese Freundin hat es an Weihnachten 2005 leider gesundheitsmäßig schwer mit etwas getroffen, das man *niemals*, und schon gar nicht mit 34 Jahren bekommen sollte. Aber Silke: Du hast schon sooft gezeigt, was man durch Selbstmotivation, Durchhalten und die Welt stets positiv Sehen alles erreichen kann.

Und deswegen schaffst Du das dieses Mal auch.

