

2.6 Neu in VB2008

Die folgenden Aufzählungen fassen geordnet nach Themen die wichtigsten Neuerungen im Vergleich zu VB2005 zusammen. Noch mehr Details finden Sie am Beginn jedes Kapitels, wo in einer kurzen Zusammenfassung alle neuen Funktionen beschrieben sind.

Entwicklungsumgebung

- ▶ **Edit und Continue:** Zu den größten »Neuerungen« in VB2005 zählte die Wiedereinführung von *Edit und Continue*. Diese Funktion erlaubt die Veränderung des Programmcodes während der Ausführung. *Edit und Continue* war in VB1 bis VB6 eine Selbstverständlichkeit, kam aber in VB.NET unter die Räder und wurde erst in VB2005 neuerlich eingeführt (wenn auch mit vielen Einschränkungen).

Wer VB2008 unter einem 64-Bit-Betriebssystem nutzt, wird feststellen, dass *Edit und Continue* schon wieder nicht funktioniert. Diese Einschränkung gilt aber zumindest nur für 64-Bit-Kompilate. Sobald Sie in den Projekteigenschaften im Dialogblatt KOMPILIEREN | ERWEITERTE KOMPILIERUNGSOPTIONEN als ZIEL-CPU explizit x86 angeben, funktioniert *Edit und Continue* wieder. Unklar bleibt, warum Microsoft es nicht geschafft hat, diese Funktion auch für 64-Bit-Kompilate zu realisieren.

- ▶ **IntelliSense:** Die Eingabehilfe *IntelliSense* funktioniert nun nicht nur für Klassen-, Eigenschafts- und Methodennamen, sondern auch für die Schlüsselwörter der Sprache Visual Basic (z.B. *Structure* oder *While*) sowie in manchen Fällen für die Elemente und Attribute von XML-Daten.

Visual-Basic-Sprachelemente

- ▶ **If(a, b, c):** Diese neue Funktion ist eine Variante zu *IIf*. Wenn der Ausdruck *a* wahr ist (*True*), liefert die Funktion *b*, sonst *c*. Der Unterschied zu *IIf* besteht darin, dass nur *b* oder *c* ausgewertet wird, nicht wie bisher beide Ausdrücke (auch der für das Ergebnis gar nicht benötigte).
- ▶ **Implizite Typendeklaration:** Sie können nun Variablen auch ohne Typangabe deklarieren, wenn Sie Startwerte angeben. Visual Basic entscheidet sich selbst für einen geeigneten Datentyp. Das reduziert oft den Tippaufwand, teilweise aber auch die Lesbarkeit und Exaktheit des Codes.
- ▶ **Nullable-Datentypen:** Eigentlich gibt es Werttypen, die zusätzlich den Wert *NULL* annehmen können, bereits seit VB2005 bzw. seit dem .NET-Framework 2.0. Die erste Implementierung gelang allerdings nicht besonders gut. Deswegen hat Microsoft nochmals nachgebessert und zur Deklaration derartiger Variablen auch gleich eine kompaktere Syntax entwickelt (*Dim i? As Integer*).

- ▶ **Erweiterungsmethoden:** Sogenannte *extension methods* erlauben es, eine vorhandene Klasse, deren Code nicht vorliegen muss, von außen durch Methoden zu erweitern.
- ▶ **Partielle Methoden:** In *partial methods* können die Codegeneratoren von Visual Studio die Schablone einer Ereignisprozedur definieren. Die tatsächliche Implementierung der Prozedur ist optional.
- ▶ **Lamba-Funktionen:** Damit können Sie einfache Funktionen übergeben bzw. nutzen, ohne sie vorher explizit zu deklarieren.
- ▶ **LINQ:** Dank *Language Integrated Queries* können Sie Abfragen für Felder, Aufzählungen, Datenbanken oder XML-Daten in einer SQL-ähnlichen Syntax direkt im VB-Code angeben.

.NET-Framework 3.5

VB2008 basiert auf dem .NET-Framework 3.5. Alle Neuerungen im Framework können daher in VB2008 genutzt werden. Die in den weiteren Abschnitten beschriebenen Punkte sind daher eigentlich nicht neu in VB2008, sondern neu in .NET 3.5. Dieser Abschnitt gibt nur einen kurzen Überblick. Eine vollständige Beschreibung ist angesichts der unüberschaubaren Klassenanzahl unmöglich. Bereits das .NET Framework 1.0 enthielt mehr als 2000 Klassen, mittlerweile sind es weit über 10.000!

VB2008 ist gleichzeitig die erste VB-Version für Windows Vista. Natürlich war es schon mit VB2005 möglich, Vista-Programme zu entwickeln, aber erst VB2008 bietet direkten Zugang zu den wichtigsten Neuerungen in Vista, die bereits in die Version 3.0 des .NET-Frameworks integriert wurden. Dazu zählen unter anderem:

- ▶ das neue Grafik- und Fenstersystem (*Avalon* alias *Windows Presentation Foundation*)
- ▶ die neue Service-Infrastruktur (*Indigo* alias *Windows Communication Foundation*)
- ▶ die neue Kommunikationsinfrastruktur (*Windows Workflow Foundation*)

Das .NET-Framework 3.5 enthält im Vergleich zu .NET 3.0 nur wenig fundamentale Neuerungen: Dazu zählen die Unterstützung des Abfragemechanismus LINQ (siehe unten) sowie diverse Funktionen, die für das Zusammenspiel mit der neuen Entwicklungsumgebung Visual Studio 2008 erforderlich waren.

> > > HINWEIS

.NET 3.5 setzt die Windows-Versionen XP SP2, 2003 Server, Vista oder neuer voraus! Windows XP ohne SP2, Windows 2000 sowie ältere Windows-Versionen werden nicht mehr unterstützt.

Mit VB2008 können Sie auch Programme für die .NET-Framework-Versionen 2.0 und 3.0 entwickeln. Dann stehen Ihnen natürlich die neuen Funktionen aus der Framework-Version 3.5 nicht zur Verfügung. Dafür können Sie aber je nach Betriebssystem auf die Auslieferung des .NET-Frameworks verzichten bzw. eine Ausführung unter älteren Windows-Versionen überhaupt ermöglichen.

Windows-Programmierung (Windows Presentation Foundation = WPF)

Wie bisher können Sie Windows-Programme auf der Basis von Windows Forms entwickeln. Die entsprechenden Klassenbibliotheken sind im Vergleich zu VB2005 nahezu unverändert geblieben. Alternativ dazu bietet VB2008 die Möglichkeit, Windows-Anwendungen auf ein neues Fundament zu stellen, die *Windows Presentation Foundation* (kurz WPF). Sowohl die Steuerelemente an sich als auch die zur Programmierung erforderlichen Klassenbibliotheken wurden im Rahmen von Windows Vista vollkommen neu entwickelt.

Microsoft wird nicht müde, die Vorteile der WPF zu kommunizieren:

- ▶ Die WPF greift direkt auf das Grafiksystem DirectX zu und ermöglicht daher zahlreiche neue 2D- und 3D-Effekte in hoher Geschwindigkeit.
- ▶ Das Design von WPF-Programmen, also die Anordnung der Steuerelemente und die Einstellung ihrer Eigenschaften, erfolgt in einer XML-Datei in einem neuen, von Microsoft entwickelten Format, der *eXtensible Application Markup Language* (kurz XAML). Damit wird das Layout von Fenstern bzw. Dialogen sprachunabhängig gespeichert. Das erleichtert den Codeaustausch zwischen VB, C# und anderen .NET-Sprachen und führt zu einer klareren Trennung zwischen Layout und Code.
- ▶ Die WPF ist die Zukunft der Windows-Entwicklung, Windows Forms ist alter Code, der zwar noch gewartet, aber nicht mehr erweitert wird.

Natürgemäß ist die WPF aber auch mit Nachteilen verbunden:

- ▶ Vorhandene Windows-Forms-Programme müssen vollkommen neu entwickelt werden. Es gibt keine Migrationsmöglichkeit.
- ▶ Die Palette der WPF-Steuerelemente ist noch nicht so reichhaltig wie die für Windows Forms. Insbesondere mangelt es an leistungsstarken Steuerelementen zur Darstellung von Listen und Tabellen. Es ist zwar möglich, in einem WPF-Programm auch Windows-Forms-Steuerelemente einzusetzen, das ist aber mit vielen Nachteilen verbunden.
- ▶ Der WPF-Designer bietet weniger Funktionen und Komfort als dessen Windows-Forms-Variante und wirkt unausgereift. Immerhin können erfahrene WPF-Entwickler XAML-Code direkt eingeben. Das ist im Zeitalter grafischer Benutzerschnittstellen zwar ein Rückschritt, dafür ist die Effizienz aber sehr hoch.
- ▶ WPF-Programme setzen zumindest Windows XP SP 2, Windows Server 2003 oder Windows Vista voraus. Ältere Windows-Versionen werden nicht unterstützt.
- ▶ Die WPF ist momentan kein taugliches Fundament für Datenbankentwickler. Es fehlen sowohl elementare Datenbanksteuerelemente als auch WPF-kompatible Hilfsmittel in Visual Studio.

Einen ausführlicheren Vergleich zwischen beiden Techniken finden Sie in Abschnitt 16.1. Für dieses Buch habe ich mich trotz mancher Mängel dazu entschlossen, der WPF gegenüber Windows Forms den Vorzug zu geben. Aber auch wer weiterhin auf Windows Forms setzt oder vorhandene Win-

dows-Forms-Programme warten muss, wird nicht enttäuscht: Auf der beiliegenden CD finden Sie ein mehrere hundert Seiten umfassendes E-Book (eine PDF-Datei) zur Windows-Forms-Programmierung mit VB2008.

Datenbankprogrammierung (ADO.NET)

In ADO.NET an sich gibt es in VB2008 – ein wenig überraschend – keine Neuerungen. Die Datenbankklassen in .NET 3.5 basieren unverändert auf den aus .NET 2.0 stammenden Bibliotheken. Das ursprünglich als »ADO.NET 3.0« angekündigte *ADO.NET Entity Framework* ist nicht rechtzeitig fertig geworden und soll voraussichtlich ab Sommer 2008 – wenn auch der SQL Server 2008 fertig ist – als nachinstallierbare ADO.NET-Erweiterung zur Verfügung stehen. Die wenigen Neuerungen für Datenbankprogrammierer sind daher in den Randbereichen angesiedelt:

- ▶ **LINQ:** *LINQ to Dataset* und *LINQ to SQL* machen die oben schon vorgestellten *Language Integrated Queries* kompatibel zu ADO.NET.
- ▶ **SQL Server Compact Edition:** Diese Minimalversion des SQL Servers ermöglicht eine unkomplizierte Entwicklung und Weitergabe von Datenbankanwendungen. Die einzige wesentliche Einschränkung besteht darin, dass nur ein Benutzer auf die Daten zugreift (also kein Client-Server-Szenario). Gegenüber ihrer nächst größeren Schwester, der SQL Server Express Edition, zeichnet sich die Compact Edition durch den winzigen Weitergabe-Overhead (ca. 2 MByte) aus. Außerdem erfolgt der Datenbankzugriff direkt über eine einzige Datei, die in ein lokales Verzeichnis installiert werden kann und dann bis zu 4 GByte groß werden darf. Gleichzeitig ist die Compact Edition in ihren Grundfunktionen kompatibel zu den »richtigen« SQL-Server-Versionen.

Mit der SQL Server Compact Edition bietet Microsoft erstmalig eine praktikable Lösung für kleine Datenbankprogramme an. Die neuen *Sync Services for ADO.NET 2.0* erlauben zudem in bestimmten Fällen eine einfache Synchronisation zu »richtigen« SQL-Server-Versionen.

XML

Um XML und LINQ miteinander zu kombinieren, hat Microsoft unter dem Namen *LINQ to XML* eine neue Familie von XML-Klassen geschaffen (übrigens bereits die vierte im .NET-Framework). In diesem Zusammenhang profitieren Sie von zwei VB-Syntaxerweiterungen, die den Umgang mit XML-Daten erheblich erleichtern:

- ▶ **XML-Literale:** Zuweisungen an XML-Objekte führen Sie ganz einfach in der Form *xelement = <abc> ... </abc>* oder *xdocument = <?xml version="1.0"?><abc>...</abc>* durch, wobei der XML-Code ohne das VB-typische Zeichen `_` über mehrere Zeilen reichen darf.
- ▶ **XML-axis-Eigenschaften:** Zum Zugriff auf XML-Elemente, -Attribute etc. können Sie die besonders kompakten Schreibweisen *object.<element>*, *object.@attribut* und *object...<descendant>* verwenden. Wenn die XML-Struktur für Visual Studio bekannt ist – z.B. durch eine *.xsd-Datei des Projekts –, funktioniert für diese Schreibweisen sogar *IntelliSense*, also die automatische Namensergänzung durch die Entwicklungsumgebung.

Webprogrammierung (ASP.NET)

Die wohl wichtigste Neuerung für Webprogrammierer ist die Unterstützung von Web-2.0-Technologien (Stichwort AJAX). Daneben gibt es unzählige weitere neue Funktionen und ASP.NET-Steuer-elemente. Da dieses Buch ASP.NET aber nicht behandelt, verzichte ich hier auch auf eine Diskussion der Neuerungen.

Kompatibilität zu VB2005

Erfreulicherweise gibt es in VB2008 keinerlei Kompatibilitätsprobleme beim Umstieg von VB2005. Nicht kompatibel sind allerdings Visual-Studio-Projekte: Wenn Sie ein mit einer früheren Visual-Basic-Version erstelltes Projekt öffnen, wird dieses automatisch in das Format von Visual Studio 2008 umgewandelt. Die Umwandlung ist zumeist problemlos, aber sie bringt es mit sich, dass Sie das Projekt mit alten Visual-Basic-Versionen nicht mehr bearbeiten können.

! ! ! ACHTUNG

Importierte Projekte haben zum Teil andere Projekteigenschaften als neue VB2008-Projekte. Das betrifft beispielsweise die Defaultimporte (VERWEISE im Projekteigenschaftendialog) sowie das Zielframework (.NET 2.0 bei importierten Projekten, aber .NET 3.5 bei neuen Projekten). Lesen Sie auch die diesbezügliche Hinweisbox auf Seite 489!

Nicht neu ...

VB2008 als Nullnummer für Datenbankentwickler zu bezeichnen, ist vielleicht ein wenig hart: Mit LINQ und der Integration der SQL Server Compact Edition gibt es ja zumindest zwei nennenswerte Neuerungen. Die können aber nicht darüber hinwegtäuschen, dass der SQL Server 2008 nicht rechtzeitig fertig wurde und weder Visual Studio noch ADO.NET für diese neue Version des Datenbankservers optimiert sind. Natürlich hindert Sie niemand, VB-Programme für den SQL Server 2008 zu entwickeln, sobald dieser verfügbar ist. Aber die integrierten Visual Data Tools, der Setup-Assistent, die Dokumentation etc. – alles ist auf den SQL Server 2005 ausgerichtet. Mit der Ausnahme von Datenbankprofis werden somit viele bei der alten Version bleiben bzw. nach einem Update den SQL Server 2008 so einsetzen, als hätte dieser keine neuen Funktionen ...

Der zweite ganz große Mangel für Datenbankentwickler betrifft die WPF: In der gegenwärtigen Form ist die Entwicklung ernsthafter WPF-Datenbankanwendungen extrem aufwendig. Es fehlen elementare Datenbanksteuerelemente (*DataGridView* und *BindingNavigator*), die in Visual Studio integrierten Visual Data Tools und der *DataSet*-Designer sind zur WPF inkompatibel etc. Datenbankentwickler werden so gezwungen, bis auf Weiteres bei Windows Forms zu bleiben und auf »tote« Technologie zu setzen.

Der dritte Punkt ist das *ADO.NET Entity Framework*: Diese Erweiterung zu ADO.NET soll zusammen mit der LINQ-Schnittstelle *LINQ to Entities* eine neue Abstraktionsschicht zwischen den eigentlichen Daten (Tabellen) und dem Anwendungsprogramm schaffen. Die stark auf den SQL Server 2008 ab-

gestimmten Funktionen versprechen eine bessere Verknüpfung relationaler Daten mit den objekt-orientierten Sprachmerkmalen von Visual Basic. Das sollte vor allem komplexe Anwendungen übersichtlicher und leichter wartbar machen sollte. Leider wurde auch dieses Feature nicht rechtzeitig fertiggestellt. Momentan peilt Microsoft die Auslieferung für den Sommer 2008 an. Zwar soll es möglich sein, die erforderlichen Bibliotheken sowie einen Designer für VS2008 nachträglich zu installieren, tatsächlich werden die meisten Entwickler aber wohl warten, bis das *ADO.NET Entity Framework* ordentlich in die nächste .NET- und Visual-Studio-Version integriert und entsprechend dokumentiert ist.

Interessant ist, dass von vielen VB2003- und VB2005-Neuerungen in der VB2008-Dokumentation nicht mehr viel zu hören bzw. zu lesen ist: Das betrifft etwa die in VB2005 eingeführten *My*-Klassen, die nicht erweitert wurden, insbesondere nicht für die WPF-Programmierung. Anscheinend hat man inzwischen auch bei Microsoft Zweifel, dass die Idee so gut war. Dasselbe gilt für die mitgelieferten Codeausschnitte (*code snippets*).

Auch *Refactoring* war vor drei Jahren anscheinend moderner als jetzt: Ein entsprechendes Add-In ist zwar noch immer kostenlos verfügbar, eine offizielle Integration eines entsprechenden VB-Werkzeugs in die Entwicklungsumgebung ist aber nicht in Sicht. Stillstand herrscht auch beim Code-Obfuscator: Die mitgelieferte Version leidet unter denselben Schwächen wie die erste Version dieses Programms von 2003.

Zu guter Letzt noch eine Anmerkung zum MSDN-Hilfesystem: Dieses ist im Laufe der Zeit untrüglich langsam geworden. Viele Seiten entbehren jeglicher Information, wenn man einmal von der Syntaxbeschreibung absieht; diese verrät aber ohnedies der Objektbrowser. Die Übersetzung ins Deutsche ist in vielen Fällen so schlecht, dass man den Sinn bestenfall erraten kann. Eine Suche innerhalb des MSDN (*Microsoft Developer Network*) ist weitgehend sinnlos – das gesuchte Ergebnis findet sich, wenn überhaupt, gut versteckt zwischen 100 anderen MSDN-Seiten zu vollkommen anderen Themen. Ein Trauerspiel! Google führt in den meisten Fällen schneller zu relevanten Informationen.