Kapitel 1

ActiveX-Mythen

Da meinte mein Verleger doch, daß ein Kapitel mit nur so wenigen Seiten wohl kaum üblich und angemessen wäre.

Aber Länge ist schließlich relativ. So mag dieses Kapitel zwar vom Umfang her unscheinbar sein, doch es braucht sich von seiner Bedeutung her keinesfalls hinter einem der übrigen Kapitel zu verstecken.

Am Anfang des Einstiegs in eine neue Technologie ist es zunächst wichtig, die Weichen richtig zu stellen. Falsche Annahmen oder Vorurteile können das Verständnis einer Technologie und damit die korrekte Anwendung erschweren.

Das gilt in besonderem Maße angesichts der heutigen rasanten Fortentwicklungen der Technologie. Häufig beruht das erste Wissen über eine neue Technologie auf Informationen seitens des Anbieters, besonders häufig natürlich von Microsoft. Leider ändern sich diese Informationen recht häufig, und allzu oft haben Marketing-Leute die Dinge in ihrem Sinne »zurechtgerückt«. Die richtige Abschätzung neuer Technologien wird nahezu unmöglich, wenn diese mit Marketing-Schlagwörtern und Abkürzungen zugekleistert werden. So wird es ein recht aufwendiges Unterfangen, aus der verkäuferischen Informationsflut die wirklich wichtigen und interessanten Technologien herauszufiltern.

So ist es kein Wunder, daß im Feld des Marketing-Hypes und der Industriepolitik die Begriffe COM, ActiveX und OLE große Verwirrung stiften. Verschiedene Mythen sind mittlerweile so weithin verbreitet, daß sogar mein Verleger darauf drang, den Titel dieses Buches zu ändern, da doch nur wenige Programmierer an ActiveX-Technologie interessiert wären. Aber wenn Sie erst einmal Kapitel 2, »ActiveX – eine historische (aber auch technische) Betrachtung«, hinter sich gebracht haben, werden Sie feststellen, wie sehr sich diese Programmierer doch täuschen dürften. Trotzdem habe ich den Titel an diese Erwartungshaltung angepaßt, in der Hoffnung, daß vielleicht doch der eine oder andere dieser Programmierer einen Blick in dieses Buch riskiert und so die Chance zu einer anderen Perspektive erhält.

Gehören Sie zu denjenigen, für die COM und ActiveX etwas relativ Neues sind, und haben Sie noch nicht allzu viele von jenen Mythen mitbekommen? Dann sollten Sie sich glücklich schätzen – Sie werden mit weniger Vorurteilen zu kämpfen haben. Machen Sie sich nichts daraus, wenn Sie einige der Begriffe in diesem Kapitel noch nicht ganz verstehen, sie werden alle in den weiteren Kapiteln erläutert.

Doch zunächst wollen wir einige der Mythen aus dem Weg räumen, so daß wir das Studium von COM und ActiveX mit freiem Kopf beginnen können.

Mythos: COM-Komponenten haben nichts mit ActiveX zu tun – dies ist nur eine fehlgeschlagene Microsoft-Technologie.

Tatsache: COM-Komponenten *sind* ActiveX-Komponenten.

Sie können nicht über ActiveX-Komponenten sprechen, ohne gleichzeitig über COM zu reden. Jede auf COM-Technologie basierende ActiveX-Komponente ist

eine ActiveX-Komponente, ob Ihnen das gefällt oder auch nicht. Und jedes von Ihnen geschriebene Visual-Basic-Programm macht sich die Vorteile von COM- und ActiveX-Technologien zunutze – selbst wenn Sie niemals eine eigenständige Komponente entwickeln sollten, wird Ihnen das Verständnis dieser Technologien viel bringen.

Worin besteht denn nun der Unterschied zwischen COM und ActiveX? Für die Puristen unter Ihnen werde ich später noch einmal auf diese Frage zurückkommen. Aus der Sicht eines Visual-Basic-Programmierers ist beides praktisch ein und dasselbe. Sie können in diesem Buch die Begriffe COM und ActiveX einfach und beliebig austauschen.

Mythos: ActiveX ist bloß für die Internet-Programmierung und für Web-Sites von Bedeutung.

Tatsache: Active ≠ Internet

Auch wenn einige Aspekte von ActiveX sich besonders auf das Internet beziehen, hat der größte Teil der ActiveX-Technologien überhaupt nichts mit dem Internet zu tun.

Nun werden sicher einige von Ihnen überrascht aufschauen. Sie werden sagen, daß Microsoft die ActiveX-Controls doch als Ersatz für die OLE-Controls geschaffen hat. Nun, damit würden Sie lediglich meine Ansicht bestätigen, daß vorschnell gefaßte Ansichten dem Verständnis von neuen Technologien im Wege stehen. ActiveX-Controls haben nämlich ganz und gar nicht die OLE-Controls ersetzt. ActiveX-Controls *sind* OLE-Controls! Es besteht kein Unterschied, nicht im geringsten. Warum das der Fall ist, und wie es dazu gekommen ist, werden Sie im nächsten Kapitel erfahren.

Mythos: ActiveX und COM sind nur etwas für fortgeschrittene Visual-Basic-Programmierer.

Tatsache: ActiveX und COM sollten zu den ersten Dingen gehören, die ein Visual-Basic-Einsteiger lernt.

Naturgemäß wird eine Programmiersprache wie Visual Basic in der Abfolge gelernt, wie neue Features dieser Sprache hinzugefügt wurden. Ganz besonders krass fällt dies ins Auge bei vielen Büchern zu Visual Basic 4.0, in denen die seinerzeit neu eingeführten Klassen-Module gewissermaßen nur als Anhängsel vorkamen, nachdem all die anderen Features der Sprache bereits abgehandelt worden waren.

Ein schwerwiegender Fehler, wie ich meine.

Denn gerade die Klassen-Module waren das aus Programmierersicht wichtigste Feature, das mit der Version 4.0 das Licht der Welt erblickte. Zum ersten Mal konnte man mit Visual Basic wirklich objektorientiert programmieren. Zugegeben, Visual Basic war nie eine und ist immer noch keine wirklich objektorientierte Sprache im striktesten Sinne der Definition dieses Begriffs. Aber die Klas-

sen-Module ermöglichen immerhin die Anwendung einiger wesentlicher Schlüsseltechniken objektorientierter Programmierung. Ganz im Vordergrund steht die Möglichkeit, Information mit einer klar definierten Schnittstelle zu umkleiden. Somit können Daten in einer Klasse vor dem übrigen Programm verborgen und der Zugriff auf diese Daten nur durch eigens in der Klasse implementierte Funktionen zugelassen werden.

Was hat dies mit Programmier-Einsteigern zu tun?

Vergegenwärtigen Sie sich einmal, wie viele Visual-Basic-Programmierer vorgehen, wenn sie eine Anwendung entwickeln:

1. Sie zeichnen zunächst Controls auf ein Formular (legen die Benutzeroberfläche an).
2. Dann wird Code für die verschiedensten Ereignisse hinzugefügt.
3. Schließlich kommen weitere Funktionen im Allgemein-Abschnitt der Formulare oder der Module hinzu, abhängig davon, ob sie von anderen Modulen mitgenutzt werden, je nach Lust und Laune des Programmierers. Es werden Variablen definiert – und so entsteht auf diese Weise etwas, was irgendwie nach einem Programm aussieht.

Diese Methode eignet sich hervorragend für triviale Programme oder »Wegwerf«-Code. Sollte dies jedoch der Stil sein, den Sie generell beim Programmieren bevorzugen, dann möchte ich Sie dazu einladen, einmal über einen anderen Weg nachzudenken:

1. Denken Sie über die Architektur der Anwendung nach: Anforderungen, Algorithmen, Benutzeroberfläche, Datenstrukturen, Terminplanung, Ressourcen und mehr. Definieren Sie die Klassen, auf denen die Anwendung aufbauen soll.
2. Klassen bilden den Kern der Datenstrukturen und Funktionsbereiche ab. Klassen können aus anderen Klassen zusammengesetzt sein. Betrachten Sie auch die Formulare als Klassen! Weiterhin können Sie die Planung zusätzlicher Benutzeroberflächen-Elemente (ActiveX-Controls) für Ihre Anwendung in Angriff nehmen.
3. Jetzt erst legen Sie richtig los: Sie zeichnen die Formulare und implementieren das Programm.

Der Entwurf kommt immer zuerst. Nehmen Sie sich die Zeit, Ihre Anwendung zuerst zu skizzieren. Dies ist das wichtigste für qualitativ hochwertigen, schlanken, zuverlässigen und schließlich schnellen Code.

Darüber hinaus wird ein auf Klassen beruhendes Programm einfach besser. »Objektorientiertes Programmieren« ist nicht bloß Marketing-Geschwätz, das Programmiersprachen-Anbieter für ihre Anzeigen ausgeheckt haben, um den Umsatz zu vervielfachen. Sicher, darum geht es auch, aber es steckt viel mehr dahinter. Nämlich eine Entwurfstechnik und eine Implementationsmethodologie,

die Ihnen dabei hilft, besseren Code zu schreiben. Jeder Visual-Basic-Programmierer sollte das lernen, insbesondere Neueinsteiger.

Was hat das alles mit ActiveX zu tun? Ganz einfach: Wie Sie in Kapitel 2 sehen werden, sind Klassen nichts anderes als ActiveX-Komponenten. Seit der Version 4.0 wurde Visual Basic sogar selbst intern auf der Grundlage von ActiveX-Technologie programmiert.

Mythos: ActiveX kommt für Sie nur in Betracht, wenn Sie Server-Komponenten oder Controls entwickeln wollen.

Tatsache: ActiveX ist wichtig für jeden Programmierer.

Nun, wenn ActiveX, wie gesagt, die Implementierungstechnologie für Klassen ist und Klassen die Implementierungstechnologie für COM-Komponenten und für objektorientierte Programmierung darstellen und dann auch noch objektorientierte Programmierung wichtig für jeden Visual-Basic-Programmierer ist – welche Schlußfolgerung ergibt sich unweigerlich daraus? Richtig: ActiveX ist wichtig für *jeden* Visual-Basic-Programmierer. Nun, nichts anderes wollte ich Ihnen von Anfang an klarmachen.

Gestatten Sie mir jedoch, diesen Punkt noch ein wenig zu vertiefen. ActiveX ist die fundamentale Technologie der Objekte unter Visual Basic. Sie kann auf verschiedenste Weise eingesetzt werden. Daher sollten Sie zwei grundsätzliche Dinge beim Entwurf Ihrer Anwendungen berücksichtigen.

Erstens: Sie definieren zunächst das Objekt-Modell Ihrer Anwendung. Welche Objekte brauchen Sie und welche Schnittstellen treffen am besten die Anforderungen Ihrer Anwendung?

Zweitens: Sie legen fest, wie diese Objekte verwendet werden. Welche von ihnen sollen öffentlich sein, welche dagegen privat? Soll ein Objekt als Teil einer Anwendungs-Klasse oder als In-Process-Code-Komponente (ein DLL-Server), als Out-of-Process-Code-Komponente (EXE-Server), als ActiveX-Control (OLE-Control) oder als ActiveX-Dokument (DocObject) angelegt werden?

Als Visual-Basic-Programmierer haben Sie unter Visual Basic 6.0 wesentlich mehr Möglichkeiten als unter früheren Versionen. Ihnen diese Möglichkeiten vorzustellen, ist der Sinn und Zweck dieses Buches.

Fangen wir also damit an.