Roundtrip durch Visual Studio .NET 2003

Die Summe der neuen Features, die Visual Studio 1.1 mit sich bringt, ist beachtlich. Äußerlich hat sich zwar nicht viel getan, unter der Haube verbergen sich aber eine Menge Modifikationen, die sowohl die Sprachen als auch Visual Studio betreffen.

___ Ein erster Blick auf die neue Entwicklungsumgebung zeigt ein verändertes Design der Startseite, wobei die eigentliche Funktionalität beibehalten wurde. Ansonsten hat sich die Entwicklungsumgebung nicht stark verändert. In der Startseite wurde die seitliche Navigationsleiste in ein eigenes Fenster verschoben und ist nun unter *Online-Resources* zu finden. Alle Links, die auch in der Vorgängerversion zu finden waren, sind noch vorhanden.

Die Installation erfolgt reibungslos. Beide Versionen von Visual Studio laufen nebeneinander. Beim ersten Start ermöglicht es die neue Version, Einstellungen eines bereits vorhandenen Visual Studio zu übernehmen. Das freut den Anwender, mindert es doch den Aufwand für die unumgänglichen Anpassungen auf persönliche Vorlieben. Die Übernahme der Einstellungen erfolgt ebenfalls ohne Probleme.

Ein neues Tool für die Entwicklergemeinde

In verschiedenen Newsgroups und auch bei Diskussionen oder auf Konferenzen wurde nach Mitteln und Wegen gesucht, den Zugriff auf den erzeugten Code zu verhindern. Da .NET mit einem Zwischencode arbeitet (MSIL-Code), der natürlich auch dekompiliert werden kann (das bekannteste Tool hierbei ist wohl Anakrino, siehe [1]), ist das keine leichte Aufgabe. Kaum war das Problem erkannt, gab es auch schon die ersten so genannten Dotfuscatoren, die bestehenden MSIL-Code unlesbar machen sollten.

Microsoft liefert zum neuen Visual Studio ein solches Tool mit, die Community-Edition des Dotfuscators von Preemptive. Damit wird das Dekompilieren erschwert und der Urheberschutz besser berücksichtigt. In jedem Fall ein nützlicher Zusatz zum Paket.

Fest integriert ist nun auch die Sprache Visual J#. Die Liste verfügbarer Applikationen wurde um mobile ASP.NET-Seiten sowie die Entwicklung von Applikationen für Smart Devices (Pocket PC, Windows CE) erweitert. Das .NET Compact Framework ist integriert.

Verwendung unterschiedlicher Framework-Versionen

Visual Studio .NET 2003 bringt das .NET Framework in der Version 1.1 mit. Allgemein bekannt ist hierzu, dass beide Versionen nicht kompatibel sind. In eine Richtung – nämlich von 1.0 zu 1.1 – ist das kein Problem. Auf Version 1.0 basierende Programme laufen mit der Version 1.1 nach einer kleinen Änderung in der Konfigurationsdatei. Diese Vorgehensweise hat allerdings ein Problem: Ist auf dem Zielsystem nur die .NET-Version 1.0 installiert, läuft das Programm nicht mehr. Sind auf dem Zielsystem beide Versionen installiert, läuft das Programm mit Version 1.1. Waren beide Versionen auf dem Zielsystem installiert und die Version 1.1 wurde entfernt – läuft das Programm mit Version 1.0. Ist nur Version 1.1 installiert, funktioniert es ebenfalls.

In die andere Richtung ist es schwieriger. Version 1.1-Programme laufen durchaus auch mit der Version 1.0, allerdings muss dazu in der Konfigurationsdatei der Applikation jede einzelne Assembly umgeleitet werden. Diese Umleitung resultiert in einer endlosen Liste von XML-Kommandos, die in die Konfigurationsdatei eingefügt werden müssen. Dafür läuft das Programm auf jeden Fall. Ist nämlich Version 1.0 installiert, wird diese Version in jedem Fall verwendet. Ist stattdessen Version 1.1 installiert und 1.0 nicht, verwendet die Applikation Version 1.1.

Letztendlich wird es wohl darauf hinauslaufen, dass in Zukunft Version 1.0 des Frameworks nicht mehr installiert werden wird, sondern nur noch Version 1.1. Versuche haben gezeigt, dass das Konvertieren bestehender Projekte auf die neue Version unproblematisch ist und es keine Abstürze gab. Diese Konvertierung wird übrigens automatisch vorgenommen, wenn ein Projekt, das mit Version 1.0 erstellt wurde, in VS.NET 2003 geladen wird.

Sowohl C# als auch VB.NET helfen beim Erstellen der entsprechenden Konfigurationsdateien. Durch eine Einstellung in den Projekteigenschaften lässt sich eine Applikation automatisch so konfigurieren, dass sie entweder mit .NET 1.0, 1.1 oder mit beiden Framework-Versionen läuft. Dabei ist darauf zu achten, dass nicht alle Features der Version 1.1 auch von der Version 1.0 unterstützt werden. Im Falle von C# befindet sich diese Einstellung unter *Allgemein*, bei VB.NET unter *Erstellen*.

Ein Steuerelement ist auch hinzugekommen, das den Preis für das Update – sofern er so niedrig ist wie angekündigt – allemal

SUMMARY

Auf einen Blick
Die Beta des neuen Visual Studio .NET 2003 ist offiziell für MSDN-Abonnenten verfügbar. Die wichtigsten Neuerungen werden hier beschrieben.

Eingesetzte Anwendungen
Visual Studio .NET 2003 Final Beta

Autor
Frank Eller ist .NET-Programmierer der ersten Stunde und Autor für den Verlag Addison-Wesley. Sie erreichen ihn in den .NET-Newsgroups von Microsoft oder auf seiner Website www.frankeller.de.

aufwiegt. Es handelt sich um einen Dialog zur Verzeichnisauswahl, der zwar in Version 1.0 auch schon vorhanden war, dort aber so gut versteckt ist, dass man ihn kaum fand. Im Übrigen beschränken sich die Änderungen hauptsächlich auf die Programmiersprachen. Eine genauere Betrachtung verdienen die drei Sprachen, die schon in der ersten Version verfügbar waren – C++, C# und Visual Basic .NET.

Neues für Visual-C++-Programmierer

Visual C++ hat die gravierendste Änderung erfahren. Die managed Extensions für C++ sind in der ersten Version von Visual Studio nicht vorbildlich, denn an eine komfortable Entwicklung unter .NET ist nicht zu denken. Die Windows.Forms-Bibliothek kann zwar wie alle anderen .NET-Bibliotheken benutzt werden, es muss jedoch alles händisch programmiert werden. Diejenigen, die es einmal versucht (oder gar getan) haben, kennen das Problem.

Diese Zeiten sind vorbei. Visual C++ .NET erhält mit der neuen Version neue Projektarten – nämlich Windows.Forms-Projekte. Der Designer ist der gleiche wie auch bei VB.NET und C#, und damit ist es endlich möglich, auch unter C++ komfortabel Windows-Applikationen zu erstellen. Diejenigen, die dennoch weiterhin mit den MFC programmieren wollen, bekommen ebenfalls eine neue Version (7.1). Außerdem gibt es einige Änderungen bezüglich STL, ATL, oder ATL Server. Diese im Detail hier zu beschreiben würde den Rahmen sprengen. Unbedingt ansehen sollte man sich aber die Informationen des neuen Compilers, denn Microsoft hat sich stark dem C++-Standard angenähert. Dadurch kann es passieren, dass existierende C++-Applikationen sich nicht mehr kompilieren lassen. Die Gründe dafür und Möglichkeiten, dies zu umgehen, werden aber mitgeliefert.

Neu für C#-Programmierer

C# hat eine neue Präprozessor-Direktive bekommen. *#line hidden* lässt den Debugger alle Zeilen bis zur nächsten *#line*-Direktive überspringen. Für die Dokumentationskommentare ist jetzt auch eine mehrzeilige Form verfügbar, die sich an die Java-Syntax anlehnt. Ein mehrzeiliger Dokumentationskommentar beginnt mit /** und endet mit */. Der Nachteil bei dieser Art Kommentar ist allerdings, dass die XML-Tags für die eigentliche Dokumentation von Hand eingegeben werden müssen.

IntelliSense wurde für C# ebenfalls stark erweitert. Ereignisbehandlungsroutinen können beispielsweise automatisch erzeugt werden; alles was noch nötig ist, ist das Tippen des Zuweisungsoperators += an das Event und Drücken der Tabulatortaste. Visual Studio fügt automatisch eine Methode ein, die zum Ereignis passt.

Beim Ableiten von Schnittstellen kommt ebenfalls die Tabulatortaste zum Einsatz. Der Code-Editor erstellt dann automatisch Rümpfe für alle durch die Schnittstelle geforderten Methoden. Diese Erweiterung wird sehr häufig genutzt werden, denn eigentlich wurde ja immer vergessen, eine Methode zu implementieren.

IntelliSense merkt sich jetzt auch die Methoden einer Klasse, die am häufigsten genutzt werden. Ein gutes Beispiel ist die Methode *Console.WriteLine*. Eigentlich musste man immer bis zum „L" schreiben, um den Methodennamen automatisch komplettieren zu können. Sobald *WriteLine* allerdings öfter genutzt wird, springt IntelliSense bereits bei der Eingabe des „W" auf diese Methode (*Write* wird übersprungen). Das beschleunigt naturgemäß die Eingabe.

Abschließend hilft IntelliSense auch beim Überladen von Methoden. Sobald das Schlüsselwort *override* geschrieben und die Leertaste gedrückt wird, werden alle Methoden der Basisklasse angezeigt, die überladen werden können.

Neu für VB.NET-Programmierer

Die Sprache, die wohl am weitesten verbreitet ist, hat in der neuen Version nur zwei Änderungen erfahren, die allerdings interessant sind. In *For*- und *For Each*-Schleifen können Laufvariablen nun im Kopf der Schleife deklariert werden. In der Vorgängerversion musste erst die Variable deklariert und dann die Schleife angewendet werden. Folgende Konstruktionen sind damit möglich:

```
For i As Integer = 1 To 10
  'Anweisungen ...
Next
```

beziehungsweise

```
For Each s As String In MyStrings
  'Anweisungen
Next
```

Wie auch in C# ist die Variable *i* (beziehungsweise die Variable *s*) in diesem Fall nur innerhalb der Schleifenkonstruktion gültig.

Die zweite Änderung besteht in Bitshift-Operatoren (>> beziehungsweise <<), die auch aus anderen Sprachen bekannt sind.

Eine eher kosmetische Änderung, die nichts direkt mit der Sprache zu tun hat, sind Begrenzungslinien im Quellcode. Unterhalb einer Klasse wird nun einfach ein Strich gezogen, der wohl der Trennung mehrerer Klassen dienen soll, wenn diese sich in einer Datei befinden. Eigentlich sollte das aber nicht der Fall sein.

... und noch mehr

Es gibt noch einige weitere Änderungen. Der Debugger beispielsweise wurde auch umfangreichen Änderungen unterzogen, unterstützt nun Remote Debugging unter Verwendung von Pipes statt TCP/IP, sichereres Jitten und erweiterte Fehlermeldungen. Der Projektmappen-Explorer wurde überarbeitet, er folgt nun der aktuell bearbeiteten Datei. Diese wird automatisch markiert, wenn man darin arbeitet. Falls die entsprechende Datei sich in einem Unterverzeichnis befindet, wird dieses geöffnet.

Fazit

Alles in allem wird deutlich, dass Microsoft ein rundes Paket geschnürt hat, das den Preis für das Update allemal wert ist. Die zahlreichen Bugs, die in der neuen Version gefixed wurden, wurden überhaupt nicht angesprochen. Aus diesem Grund dürfen wir uns alle auf ein neues, besseres Visual Studio .NET freuen. ||||||

[1] Anakrino Decompiler: www.saurik.com/net/exemplar/
[2] .NET Framework v1.1 Beta: http://msdn.microsoft.com/netframework/productinfo/v1.1/default.asp
[3] Visual Studio .NET Roadmap: http://msdn.microsoft.com/vstudio/productinfo/roadmap.asp