

# Kompilierte Visual-Basic-Projekte debuggen

Normalerweise können Sie Visual-Basic-Projekte in der IDE komfortabel debuggen und erhalten ein fehlerfreies Programm. Sollte Ihre fertige Anwendung dennoch Fehler verursachen, greifen Sie in die Trickkiste und lassen sich von Visual C++ Debug-Informationen erzeugen, mit deren Hilfe Sie die wahre Ursache des Fehlers ermitteln können.

VB ist eine Mischung aus Interpreter und Compiler. In der IDE wird der Programmcode interpretiert und lässt sich komfortabel debuggen. Ist das Programm fertig, wird der Code kompiliert und verhält sich fast immer so wie in der IDE während des Debuggens. Es gibt jedoch Ausnahmen. In seltenen Fällen treten Fehler oder gar Abstürze nur zur Laufzeit in der EXE-Datei auf. Die Ursache für ein solches Verhalten lässt sich nur schwer herausfinden. Sie können in der Prozedur, in der Sie den Fehler beziehungsweise die Absturzursache vermuten, damit beginnen, Meldungsfenster einzufügen oder eine Art Logdatei zu füllen. Dies tun Sie bis Sie sich Schritt für Schritt an die den Fehler oder Absturz verursachende Anweisung herangetastet haben.

Ein anderes Problem tritt in seltenen Fällen beim Debuggen von ActiveX-DLL-Projekten auf. DLLs werden im Prozessraum des aufrufenden Programms

ausgeführt, also zum Beispiel im InetInfo.exe-Prozess (IIS) oder im Host-Prozess des MTS. Wenn Sie das zugehörige DLL-Projekt in Visual Basic laden, einen Haltepunkt setzen und das Debugging starten, dann passiert im Hintergrund Folgendes:

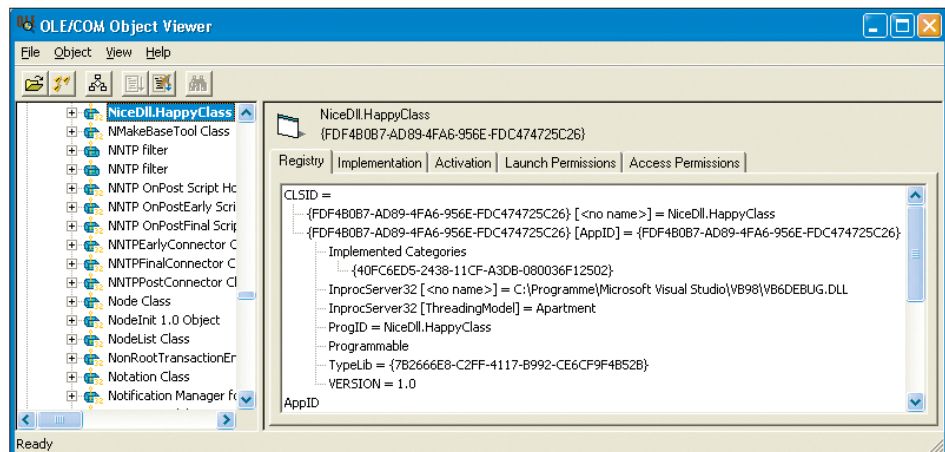


Bild 1 | Beim Debuggen von ActiveX-InProcess-Komponenten in VB wird der InprocServer32-Key in der Registry auf die VB6Debug.dll gesetzt.

VB wird den Aktivierungseintrag der COM-Komponente in der Registry (InprocServer32) von Ihrer kompilierten DLL auf die VB6Debug.dll (Bild 1) ändern.

Instanziert nun ein anderer Prozess ein Objekt aus Ihrer DLL, übernimmt die VB6Debug.dll diese Aufgabe und leitet alle Aufrufe an den VB6.exe-Prozess weiter. Dadurch wird für Prozeduraufrufe eine Interprozesskommunikation notwendig, und aus der InProcess-Komponente (DLL) wiederum wird plötzlich eine OutOfProcess Komponente (VB6.exe). Da nun eine Reihe von Sicherheitseinstellungen und Konfigurationen gelten, die zuvor irrelevant waren, kann sich das Verhalten der Komponenten erheblich ändern. Dies gilt insbesondere für die Komponenten, die im MTS ablaufen sollen.

## Hilfe aus der fremden Umgebung

In diesen Situationen schafft der Debugger von Visual Studio Abhilfe. C++-Projekte werden immer als EXE getestet. Das Bin-

## SUMMARY

### Auf einen Blick

Fehler im kompilierten VB-Code lassen sich am einfachsten mit Hilfe des C++-Debuggers finden.

### Eingesetzte Anwendungen

Visual Basic 6.0, Visual Studio 6.0

### Autor

Marcel Gnath ist Senior Consultant bei NTeam GmbH. Seine Arbeitsschwerpunkte liegen im Bereich COM, .NET und verteilte Informationssysteme, zu denen er auch Trainings durchführt. Sie erreichen ihn unter marcel@gnath.net oder www.gnath.net.

deglied zwischen dem Intel-Maschinencode und dem Programmcode ist eine Datei, die symbolische Debug-Informationen enthält. Diese Datei kann ebenso vom VB6-Compiler erzeugt werden.

Wollen Sie mit VS6 kompilierte VB-Anwendungen testen, öffnen Sie eines Ihrer VB-Projekte. Am einfachsten nachvollziehen können Sie es bei Projekten vom Typ Standard-Exe. Wie Sie COM-Komponenten debuggen, lesen Sie unter [1].

Setzen Sie bei den Projekteigenschaften auf der Seite **Kompilieren** das Häkchen bei **Debug-Informationen für symbolischen Debugger generieren** (Bild 2). Anschließend kompilieren Sie das Programm und schließen Visual Basic. Im Verzeichnis Ihres kompilierten Programms finden Sie jetzt eine zusätzliche Datei mit der Endung .pdb. Sie enthält die Quellcode-Informationen, die für den VC++-Debugger notwendig sind.

Nun starten Sie Visual C++ 6.0 und wählen im Menü **Datei** den Punkt **Arbeitsbereich öffnen**. Im Dialog ändern Sie den Dateityp-Filter auf **Alle Dateien** und öffnen Ihre soeben kompilierte Exe-Datei. Wählen Sie nun aus dem Menü **Datei** den Punkt **Öffnen**, ändern wiederum im Dialog den Dateityp auf **Alle Dateien** und wählen das VB-Formular (frm), die Klasse (cls) oder das Modul (bas) aus, das Sie debuggen möchten.

In VC++ sehen Sie jetzt den kompletten Quellcode des Moduls inklusive der Eigenschaften und Attribute. Darunter finden Sie den Quellcode. Jetzt setzen Sie an der gewünschten Stelle einen Haltepunkt. Standard-Exe-Projekte starten Sie – wie aus Visual Basic gewohnt – über die Taste [F5]. Bedienen Sie Ihr Programm und es sollte an der gewünschten Stelle anhalten. Sie müs-

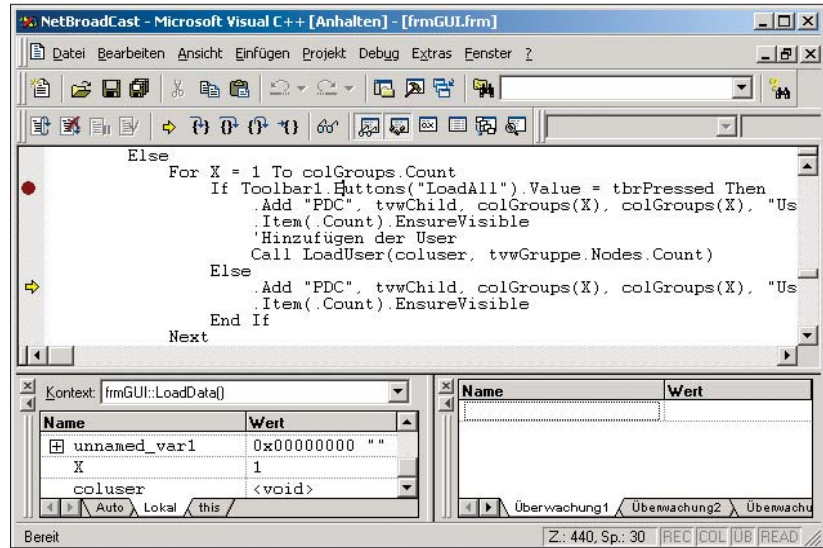


Bild 3 | So sieht ein VB Programm im C++-Debugger aus.

sen lediglich darauf achten, dass die Tastaturbelegung unterschiedlich ist. Die Tasten [F5] und [F9] haben die gleiche Funktionalität, aber Einzelschritt und Prozedurschritt entsprechen hier den Tasten [F10] und [F11].

Arbeiten Sie mit Visual Basic 5.0, können Sie genauso vorgehen und Visual C++ 5.0 zu Hilfe nehmen. |||||

[1] HowTo: Debug a Native Code VB Component in VC++ (Q166275)

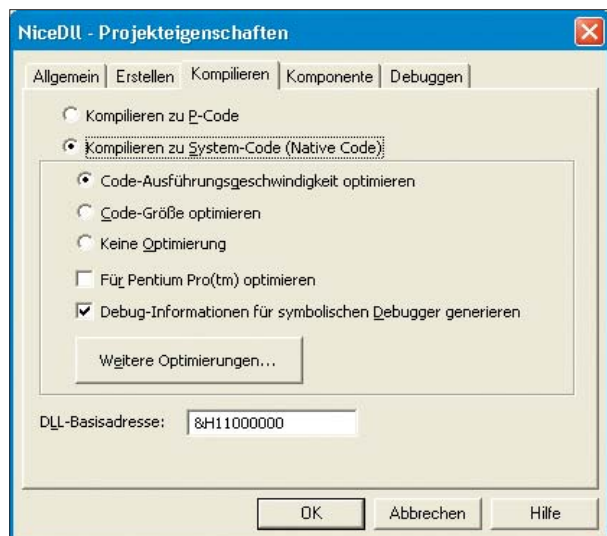


Bild 2 | Über die Projekteigenschaften legen Sie fest, ob der VB-Compiler symbolische Debugging-Informationen für den C++-Debugger generieren soll.

**AIT**

## Individual .net Software & E-Business Solutions

AIT ist ein führender Anbieter von komplexen, individuellen E-Business- und Client-Server-Anwendungen\*.

Zu unserem Leistungsportfolio gehören:

- E-Business & IT-Consulting
- Requirement-Analyse & Konzeption
- Web- & Software-Lokalisierung
- C#, .net, XML, BizTalk Entwicklung

\* auf der Basis von Microsoft .net und Windows 2000-Architektur

AIT - Applied Information Technologies AG  
 Höhenstraße 21, 70736 Stuttgart-Fellbach  
 Tel +49 (0) 711 / 52 04 73 - 10  
 Fax +49 (0) 711 / 52 04 73 - 30  
 Web www.aitag.com  
 EMail info@aitag.com

**Microsoft**  
CERTIFIED  
Partner