



Session V-VFX

Visual Extend - Mehr als ein RAD Framework

Venelina Jordanova, Uwe Habermann

Einführung

Visual Extend 10.0. Das ist schnelle Entwicklung von Datenbankanwendungen auf dem neuesten Stand der Technik. In nur 75 Minuten zaubern Venelina und Uwe eine Anwendung aus dem Nichts. Alle Features werden durch leistungsfähige Builder erstellt. Das Ergebnis ist eine Anwendung, die mit DBC und SQL Server Datenbank läuft. Neben einfachen Formularen, werden 1:N Formulare, Auswahllisten und Beziehungen zwischen Formularen erstellt.

Einführung

Visual Extend gehört seit vielen Jahren zu den leistungsfähigsten Zusatzprodukten von Visual FoxPro. Mit Visual Extend (im folgenden Text mit VFX abgekürzt) ist es möglich in wenigen Minuten den Rahmen für eine Visual FoxPro-Anwendung voll funktionsfähig zu erstellen. Wenn vor der Anwendungsentwicklung bereits eine Datenbank zur Verfügung steht, ist es ein Leichtes mit den Assistenten von VFX innerhalb kürzester Zeit Bearbeitungsformulare zu erstellen. Lernen wir die wichtigsten Eigenschaften von VFX kennen in dem wir die Arbeitsschritte zur Erstellung einer Anwendung durchgehen.

Zum Betrieb von Visual Extend 10.0 ist Visual FoxPro 9.0 erforderlich.

Installation

Nach der Installation von VFX ist es sinnvoll, das VFX-Menü in das Standardmenü von Visual FoxPro zu integrieren. Dazu ist in der Datei *Config.fpw* eine Zeile einzufügen:

```
Command = DO <VFX-Installationspfad>\builder\vfxmnu.app
```

VFX – Task Pane

Beim ersten Start von VFP nach der Installation von VFX 10.0 wird automatisch die VFX 10.0 Task Pane in die Task Pane von Visual FoxPro integriert.

Project	Description	Folder	Modified
VCM.PJX	Vcm	C:\UWE\VCM	06/16/2007 7:11:44
VFX100TEST.PJX	Vfx100test	C:\UWE\VFX100DEMOS\VFX100TEST	06/14/2007 21:06:06
VFX100TRADERS.PJX	Vfx100traders	C:\UWE\VFX100DEMOS\VFX100TRADERS	04/28/2007 19:27:22
VFPizza.PJX	Vfpizza	C:\UWE\VFX100DEMOS\VFPizza100	04/28/2007 19:24:52

Ein nützliches Tool befindet sich in der VFX 10.0 Task Pane, der Application Manager. In einer Tabelle werden Informationen über alle VFX-Projekte verwaltet. Über den VFX – Application Manager kann ein Projekt geöffnet werden. Dabei wird automatisch der aktuelle Pfad auf den Projektordner gesetzt. Außerdem kann über den VFX – Application Manager ein „Rebuild all“ durchgeführt werden. Dabei wird das Projekt komplett kompiliert. Änderungen in Include-Dateien werden dabei berücksichtigt.

VFX – Application Wizard

Eine neue Anwendung wird mit dem VFX – Application Wizard erstellt.

The screenshot shows the 'VFX - Application Wizard' dialog box at step 1. The title bar reads 'VFX - Application Wizard'. The main heading is '1. With this wizard you create a new VFX project'. Below this, there are four input fields with browse buttons (three dots): 'Master VFX home folder:' with the value 'C:\VFX\VFX100\'; 'Enter the name of the new project file:' with the value 'VFX Application 1'; 'Enter the name of the new project's folder:' with the value 'C:\Users\Uwe Habermann\Documents\VFX Projects\VFX'; and 'Database name:' with the value 'DATABASE.DBC'. A note below the first field says '(Usually you don't need to modify this path.)'. At the bottom, there are four buttons: 'Cancel', '< Back', 'Next >', and 'Finish'. The 'Next >' button is highlighted with a blue border.

Beim ersten Aufruf des Wizard wird als Sprache für die zu erstellende Anwendung die Sprache der verwendeten FoxPro-Version vorgeschlagen. Bei jedem erneuten Aufruf wird die zuletzt verwendete Sprache vorgeschlagen. Nachdem die *Finish*-Schaltfläche gedrückt wird, werden aus der leeren VFX-Musteranwendung die Dateien in den neu erstellten Projektordner kopiert und anschließend kompiliert.

The screenshot shows the 'VFX - Application Wizard' dialog box at step 3. The title bar reads 'VFX - Application Wizard'. The main heading is '3. Options'. Below this, there is a paragraph: 'The following options are general settings for your application. You can modify these settings later using the VFX Application Builder'. There are two columns of options, each with a checkbox or a dropdown menu: 'Ask to save when close:' (checked), 'Enable autoedit mode:' (checked), 'Enter on the grid means edit:' (unchecked), 'Enable hooks:' (checked), 'Use DBCX compliant products:' (unchecked), 'Copy Loader.exe to new project:' (unchecked), 'Toolbar style:' (dropdown set to 'CAppNavBar'), 'Language:' (dropdown set to 'German'), 'AutoFit grids on first load:' (unchecked), 'Enable product activation:' (unchecked), and 'Use "FirstInstall.txt" file:' (unchecked). At the bottom right, there is an 'Advanced' button. At the bottom, there are four buttons: 'Cancel', '< Back', 'Next >', and 'Finish'. The 'Next >' button is highlighted with a blue border.

Funktionsumfang der neuen Anwendung

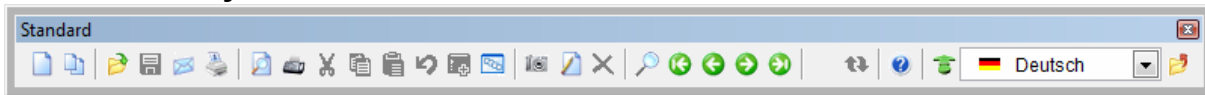
Die mit dem Application Wizard erstellte Anwendung kann sofort getestet werden. Dazu kann direkt aus dem Projekt-Manager das Hauptprogramm *Vfxmain.prg* gestartet werden. Wahlweise kann auch eine App- oder Exe-Datei erstellt und getestet werden. Dies ist während der Entwicklung normalerweise aber nicht erforderlich.

Die Anwendung startet mit einem Splashscreen. Als Bild für den Splashscreen wird eine Png-Datei verwendet, die der Entwickler leicht bearbeiten oder austauschen kann. Es ist möglich die Anzeige des Splashscreen zu unterdrücken. Nach Anzeige des Splashscreens baut sich der Hauptbildschirm auf und es erscheint der Anmeldebildschirm. Standardmäßig muss sich jeder Benutzer einer VFX-Anwendung mit einem Namen und einem Kennwort anmelden. Es ist möglich den Anmeldebildschirm zu umgehen und den Benutzer automatisch mit dem Windows-Anmeldenamen anzumelden.

Bedienung

Nach der Anmeldung wird die VFX-Anwendung ähnlich den Office-Anwendungen bedient. Benutzer, denen die Bedienung von Word oder Excel geläufig ist, können mit einer VFX-Anwendung praktisch sofort produktiv arbeiten.

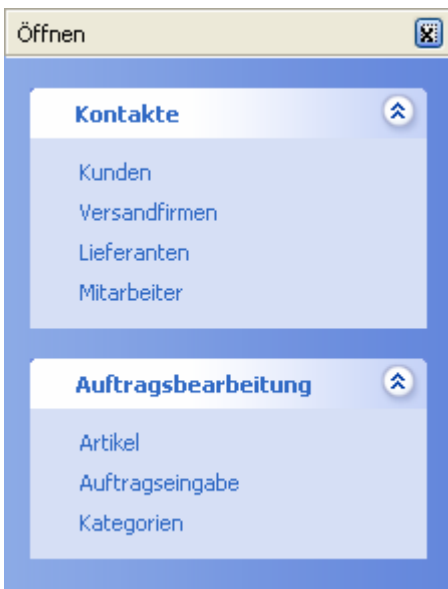
Standard-Symbolleiste



Viele der Schaltflächen der Symbolleiste sind in ihrer Funktion mit denen aus Office-Produkten identisch.

XP Öffnen-Dialog

Formulare werden standardmäßig über den Öffnen-Dialog gestartet. Der Öffnen-Dialog erscheint im Windows XP-Layout. Die Informationen der Formulare, die im Öffnen-Dialog angezeigt werden, stehen in der Tabelle *Vfxfopen.dbf*.



Formulare

The screenshot shows a window titled 'Kunden' with a tabbed interface. The 'Dateneingabe' tab is active, displaying a form with the following fields and values:

Kundennummer:	ALFKI
Firma:	Alfreds Futterkiste
Kontaktperson:	Maria Anders
Position:	Sales Representative
Adresse:	Obere Str. 57
Ort:	Berlin
Region:	
PLZ:	12209
Land:	Germany
Telefon:	030-0074321
Fax:	030-0076545
Maximum:	6300,0000
Minimum:	2600,0000
% Rabatt:	2

Wenn für ein Formular die *!Autoedit*-Eigenschaft auf *wahr* eingestellt ist (das ist der Standardwert), sind ständig alle Steuerelemente auf dem Formular aktiviert. Der Anwender kann mit der Maus oder der Tastatur ein Steuerelement anwählen und sofort mit dem Bearbeiten der Daten beginnen. Das Formular wechselt automatisch in den Bearbeitungsmodus, sobald Daten interaktiv verändert werden.

Auf der Listenseite von VFX-Formularen befindet sich ein Grid. Standardmäßig kann in allen Spalten des Grid inkrementell gesucht werden. Dazu ist einfach der Fokus in die gewünschte Spalte zu setzen. Mit dem ersten Buchstaben- oder Zifferndruck wird die Sortierfolge auf diese Spalte umgestellt. Dabei wird bei Bedarf automatisch eine temporäre Indexdatei erstellt. Die Überschrift in der Spalte wird mit einem auf- oder absteigenden Pfeil, ähnlich der Darstellung im Windows-Explorer, gekennzeichnet.

Standardmäßig kann die Größe von VFX-Formularen vom Anwender zur Laufzeit geändert werden. Alle Steuerelemente werden dabei proportional in der Größe geändert. Innerhalb von Grids wird die Größe der Steuerelemente standardmäßig nicht verändert. Wenn ein Formular vergrößert wird, werden also mehr Zeilen und Spalten im Grid sichtbar.

Alle Einstellungen an Formularen werden benutzerspezifisch gespeichert. Wenn der Anwender das Formular erneut öffnet, erscheint das Formular an der Position des Bildschirms und in der Größe in der es zuletzt geschlossen wurde. Auch die Einstellungen der Grids (Spaltenbreiten, Spaltenfolge und Sortierung) werden gespeichert.

VFX-Formulare haben normalerweise eine private Datensitzung und können problemlos mehrfach geöffnet werden. Über eine Eigenschaft des Formulars (*!Multiinstance*) kann der mehrfache Aufruf verhindert werden.

Benutzerverwaltung

In VFX ist eine Benutzerverwaltung enthalten. Dazu gehören ein Formular zur Bearbeitung der Benutzerdaten, ein Formular zur Bearbeitung der Benutzerrechte, eine Verwaltung von Benutzergruppen sowie ein Anmeldebildschirm.

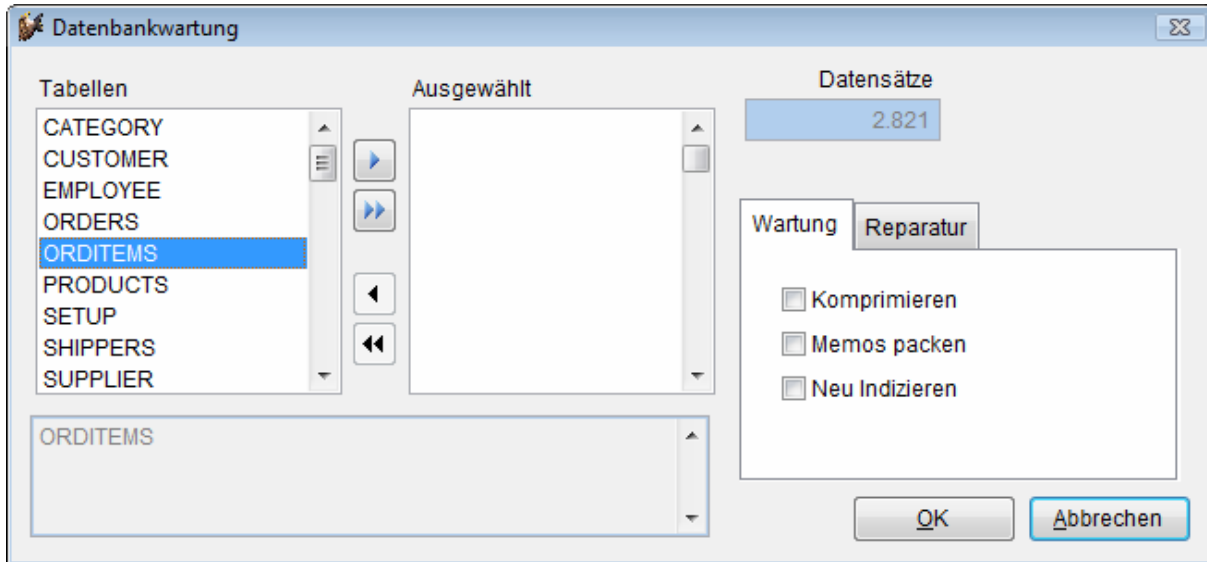
Nach der erfolgreichen Anmeldung eines Benutzers wird ein global sichtbares Objekt mit dem Namen *goUser* angelegt. Für alle Felder des aktuellen Benutzer-Datensatzes (aus der Tabelle *Vfxusr.dbf*) der dem angemeldeten Benutzer gehört, wird dem Objekt *goUser* eine Eigenschaft hinzugefügt. Der Name der Eigenschaft entspricht dem Namen des Feldes in der Tabelle *Vfxusr.dbf*. Es ist an jeder Stelle im Programm möglich, den Wert dieser Eigenschaft abzufragen um zu entscheiden, ob ein Benutzer eine bestimmte Aktion ausführen darf. So kann z. B. die Auswahl eines Menüpunkts, das Öffnen eines Formulars oder das Bearbeiten eines Feldes auf einem Formular verhindert werden.

Fehlerprotokoll

Sollte es einmal zu einem Laufzeitfehler kommen, wird der Fehler in einer MessageBox angezeigt. Außerdem wird der Fehler in einer Tabelle protokolliert. Dabei werden der Name des aktuellen Benutzers, Datum, Uhrzeit, der Status aller geöffneten Tabellen sowie die Ausgabe von List Memory gespeichert. Weitere Eigenschaften der Behandlung von Laufzeitfehlern können über Eigenschaften des Anwendungsobjekts eingestellt werden.

Datenbankwartung

Über den Menüpunkt System – Datenbankwartung wird ein Formular mit einem Mover-Dialog angezeigt.



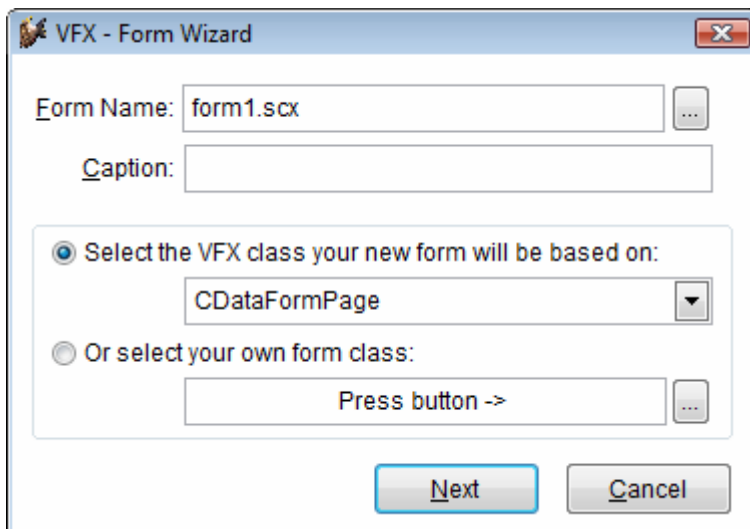
Hier können Tabellen gepackt oder indiziert werden.

Info-Dialog

Ein Standard-Info-Dialog ist in allen VFX-Anwendungen enthalten. Die angezeigten Parameter stammen aus einer Include-Datei, die beim Anlegen des Projektes erzeugt wurde.

Erstellen eines Formulars mit dem VFX – Form Wizard

Mit Hilfe des VFX – Form Wizard wird ein neues Formular auf der Basis einer VFX-Formularklasse angelegt und in das Projekt eingetragen. Die am häufigsten verwendete Formularklasse ist die Klasse *CDataFormPage*.



VFX – Data Environment Builder

Im nächsten Schritt wird in jedem VFX Form Builder die Datenumgebung bearbeitet. Die von dem Formular zu verwendenden Tabellen oder Ansichten sind in der Datenumgebung einzutragen.

Der Datenumgebung können Tabellen, Ansichten oder bestehende CursorAdapter-Klassen hinzugefügt werden oder auch neue CursorAdapter-Klassen erstellt werden. Mit einem Klick auf die Schaltfläche *Add* können bestehende Tabellen oder Ansichten der Datenumgebung hinzugefügt werden. Der VFP-Dialog zur Auswahl von Tabellen und Ansichten wird geöffnet. Wenn ein Cursor in der Datenumgebung auf einer Tabelle basiert, kann in der Spalte *Order* ein Index der Tabelle gewählt werden. Wenn ein Cursor in der Datenumgebung auf einer CursorAdapter-Klasse basiert und für diesen CursorAdapter Indexschlüssel definiert wurden, kann aus diesen Indexschlüsseln in der Spalte *Order* ebenfalls ausgewählt werden.

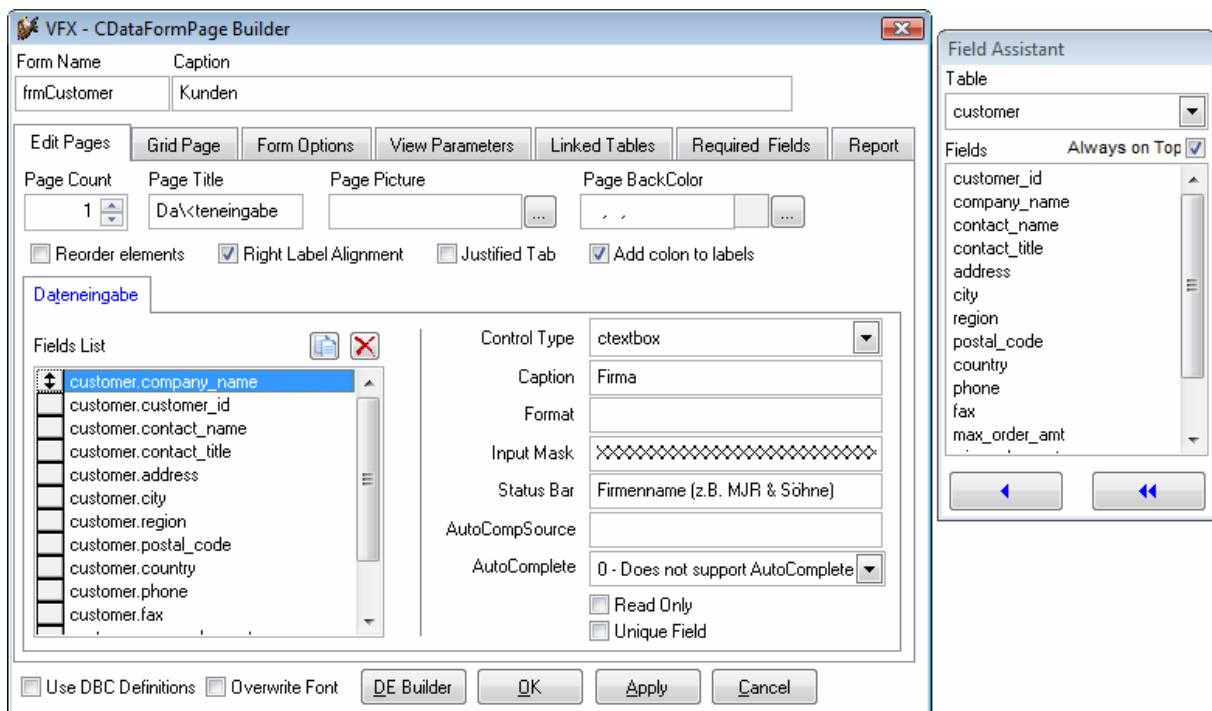
Für ein einfaches Formular zur Bearbeitung von Daten aus einer Tabelle ist es ausreichend diese Tabelle der Datenumgebung hinzuzufügen. Anschließend können dem Formular mit dem VFX – Form Builder Steuerelemente hinzugefügt werden.

Der VFX – Form Builder liest die Datenumgebung aus und stellt die Felder der Tabellen zur Auswahl um Steuerelemente zu erstellen. Zur Laufzeit wird die Datenumgebung ebenfalls ausgelesen um die Tabellen zu ermitteln, für die ein *Tableupdate* bzw. *Tablevert* durchgeführt werden muss.

Der VFX – Form Builder

Mit dem Form Builder werden die für das Formular benötigten Steuerelemente erstellt. Für jedes Steuerelement können dabei die zugrunde liegende VFX-Klasse gewählt sowie viele Eigenschaften eingestellt werden.

Beim ersten Erstellen des Formulars wird automatisch ein Eintrag in der Tabelle *Vfxfopen.dbf* angelegt, sodass das Formular über den Öffnen-Dialog gestartet werden kann.



Der VFX – Form Builder ist voll reentrant. Das heißt, man kann den Builder beliebig oft aufrufen um Einstellungen an einem Formular zu verändern. Es ist auch möglich das Formular von Hand mit VFP zu bearbeiten und anschließend wieder mit dem Form Builder zu arbeiten, ohne dass Einstellungen verloren gehen oder überschrieben werden.

Der VFX – CGrid Builder

Sollen nur Änderungen am Grid vorgenommen werden, braucht nicht der Form Builder verwendet zu werden. Mit dem VFX – Grid Builder können die Einstellungen des Grids verändert werden. Wie alle VFX Builder ist auch der Grid Builder reentrant.

Test

Das Formular kann direkt aus dem VFP Formular-Designer oder aus dem Projekt-Manager gestartet und getestet werden. Im *Init()*-Ereignis aller VFX-Formulare wird geprüft, ob das Anwendungsobjekt existiert. Falls dieses nicht vorhanden ist, wurde das Formular direkt aus dem Projekt-Manager gestartet und VFX stellt selbstständig die Umgebung her, um das Formular laufen zu lassen. Dabei wird auch die Hauptsymbolleiste instanziiert und kann für die Bedienung des Formulars verwendet werden.

Natürlich ist es auch möglich das Projekt über das Hauptprogramm *Vfxmain.prg* zu starten. Das Formular kann dann über den Öffnen-Dialog gestartet werden.

Neu in VFX 10.0

Zahlreiche neue Features sind im Laufe dieses Jahres in Visual Extend integriert worden. Die wichtigsten neuen Features sind hier beschrieben.

Unterstützung ostasiatischer Sprachen

In VFX 10.0 werden die Sprachen traditionelles und vereinfachtes Chinesisch sowie Japanisch unterstützt. Voraussetzung für die Verwendung dieser Sprachen ist eine Windows Version, die DBCS Zeichensätze unterstützt.

Config.vfx

Unterstützung des CSV Formats

Die Datei *Config.vfx* kann jetzt wahlweise im XML oder im CSV Format gespeichert werden. Dadurch ist der Einsatz auch in Umgebungen möglich, in denen MSXML nicht installiert ist und auch nicht installiert werden kann.

Dieses Verhalten wird durch die Eigenschaft *goProgram.nConfigVfxFormat* des Anwendungsobjekts gesteuert. Mit dieser Eigenschaft kann das Format festgelegt werden, in dem die Datei *Config.vfx* gespeichert wird. Beim Lesen der Datei wird das Format automatisch erkannt. Die Datei *Config.vfx* wird in jedem Fall verschlüsselt gespeichert. Zur Ver- und Entschlüsselung wird das Kennwort verwendet, das der Eigenschaft *goProgram.cConfigPassword* zugewiesen ist.

Wichtig: Wenn mit dem CSV Format gearbeitet wird, werden Zeichenketten auf eine Länge von 254 Zeichen gekürzt. Dies ist insbesondere bei der Verwendung von langen Ordnernamen und Verbindungszeichenfolgen zu beachten.

Neue Funktionen:

ReadConfigVFXtoCursor(tcFileContents, tcCursorName)

CursorToConfigVFX(tcConfigVFXFormat, tcCursorName, tcFilePath, tcPass)

CSVStringToCursor(tcDataString, tcCursorName) – Konvertiert eine Zeichenkette aus dem CSV Format in einen Cursor. Voraussetzung ist eine CSV Zeichenkette mit Kopfzeile mit Strukturinformationen.

CursorToCSVString(tcAlias) – Konvertiert einen Cursor in eine Zeichenkette im CSV Format mit einer zusätzlichen Kopfzeile mit Strukturinformationen. Felder vom Typ Memo werden in eine Zeichenkette mit maximal 254 Zeichen umgewandelt.

Datenzugriff bearbeiten

Der Menüeintrag „Datenzugriff bearbeiten“ steht nur noch Endbenutzern mit der Benutzerstufe 1 zur Verfügung.

VFX Builder

In allen VFX Buildern kann jetzt mit einem Mausklick in Grids eine neue Zeile eingefügt werden.

VFX – Grid Builder

Bei der Generierung von Spaltenbreiten wird jetzt die Breite der Überschrift berücksichtigt. Der Maximalwert von der Feldbreite und der Breite der Überschrift wird als Spaltenbreite verwendet.

Die Spaltenbreite wird durch multiplizieren von `TEXTWIDTH(...)` mal der Schriftgröße, geliefert von `FONTMETRIC(...)`, berechnet.

In Grids verwendete VFP Basisklassen bekommen vom VFX - Grid Builder die Eigenschaften für den verwendeten Zeichensatz, der in der Klassenbibliothek `Vfxobj.vcx` für die entsprechenden Klassen eingetragen sind.

Die Eigenschaft `cfixcolumnlist` von Grids wird entsprechend der `ReadOnly` Eigenschaft für Spalten von den VFX Buildern reentrant gesetzt.

VFX – Upsizing Wizard

Mit einer neuen Schaltfläche kann eingestellt werden, dass in allen Tabellen und allen Feldern (soweit möglich) der Zustand NULL zugelassen wird.

VFX – Data Environment Builder

Für Cursoradapter ist jetzt eine zusätzliche Schaltfläche für die Eigenschaft `SendUpdates` vorhanden.

VFX – Audit Trigger Wizard

Es werden nur noch Tabellen mit Primärschlüssel vorgeschlagen. Mit der Funktion `DBGETPROP(<tablename>, "TABLE", "PrimaryKey")` wird geprüft, ob ein Primärschlüssel vorhanden ist. Beim Start werden die aktuellen Einstellungen aus den Tabellen gelesen.

Es gibt neue Schaltflächen um alle Tabellen auszuwählen oder abzuwählen.

VFX – Class Switcher

Beim Wechsel von Steuerelementen zu Container-Steuerelementen wird jetzt die `ControlSource` an das Steuerelement im Container weitergegeben.

Beim Wechsel zwischen Klassen, von denen eine oder beide Containerklassen sind, werden die folgenden Werte von Eigenschaften weitergegeben: `ControlSource`, `InputMask`, `Format` und `StatusBarText`.

Öffnen-Dialog und XP Öffnen-Dialog

Der Öffnen-Dialog und der XP-Öffnen-Dialog verwenden jetzt die gleiche Funktion um Formulare nach individuellen Regeln auszuschließen. Es wird die Funktion `LoadVFXFopen()` aus `Vfxfunc.prg` verwendet, die die Daten aus der Tabelle `Vfxfopen` entsprechend der Benutzerstufe des aktuellen Benutzers liest.

XP-Öffnen-Dialog auf Terminalserver

Wenn eine VFX 10.0 Anwendung als Client auf einem Terminalserver läuft, wird die Eigenschaft `Auto Hide` automatisch abgeschaltet. In einer Terminalserver Sitzung ist im Anpassen Dialog die Eigenschaft „Öffnen-Dialog automatisch ausblenden“ nicht sichtbar.

Eindeutige Felder

Wenn versucht wird einen nicht eindeutigen Wert in einem eindeutigen Feld zu speichern, werden die Eigenschaften dieses Feldes entsprechend der Einstellung für „required field failure properties“ eingestellt, bevor eine Messagebox angezeigt wird.

Die Klasse *cDateTime*

Als *ControlSource* für die Klasse *cdatetime* werden jetzt auch Eigenschaften unterstützt.

Der Typ der *ControlSource* wird bei der Initialisierung des Steuerelements ermittelt und in der intern genutzten Eigenschaft *nControlSourceMode* gespeichert: 1 (Standardwert) Feld, 2 Variable oder Eigenschaft. Der Wert der Eigenschaft *nControlSourceMode* wird beim Speichern geprüft, um den Wert der *ControlSource* zuzuweisen oder einen *Replace* Befehl auszuführen.

Benutzerverwaltung

Das neue Feld *useraccess* in der Benutzertabelle erlaubt ein Überschreiben von Benutzergruppenrechten.

In der Benutzertabelle *Vfxusr* ist jetzt das Feld *Vfxusr.idvfxusr* für einen Primärschlüssel vorgesehen. Dieses Feld hat den Typ Integer Autoinc.

Alternativ zum bereits vorhandenen Benutzernamen kann zusätzlich in der Tabelle *Vfxusr* der Windows Anmelde-name im Feld *ntlogon* gespeichert werden.

Löschmarkierung

Anstatt Datensätze zu löschen kann in VFX 10.0 Anwendungen ein Feld vorgesehen werden, in dem eine Löschmarkierung gesetzt wird. Durch diese neue Eigenschaft ist es möglich Datenbestände mit anderen Datenbeständen zu synchronisieren. Diese Funktionalität wird in der Cursoradapter Klasse von VFX bereitgestellt.

Mit dieser Eigenschaft wird beim Löschen eines Datensatzes dieser nicht physikalisch aus der Datenbank gelöscht, sondern es wird eine Löschmarkierung gesetzt. Wenn ein Cursoradapter mit Daten gefüllt wird, werden nur Datensätze geladen, deren Löschmarkierung nicht gesetzt ist. Die Löschmarkierung wird in einem Tabellenfeld gespeichert.

Dieses Verhalten wird durch die Eigenschaft *goProgram.cDel fld* des Anwendungsobjekts gesteuert. In dieser Eigenschaft kann der Name eines Tabellenfeldes gespeichert werden. Wenn hier ein Feldname angegeben ist, wird dieser Name in allen Tabellen verwendet. Wenn der Wert dieser Eigenschaft leer ist, werden Datensätze auf dem herkömmlichen Weg gelöscht.

Der Datentyp des Feldes mit der Löschmarkierung muss N(1) sein. Als Werte werden gespeichert: 0 – Datensatz nicht gelöscht, 1 – Datensatz gelöscht.

Das Feld, welches als Löschmarkierung verwendet wird, muss in den Eigenschaften *CursorSchema*, *SelectCmd*, *UpdateableFieldList* und *UpdateNameList* des Cursoradapters angegeben werden, damit dieses Verhalten genutzt werden kann.

Eigenschaften

cFoxApp.cDel_Fld (vfxappl.vcx) – Enthält den Namen des Feldes mit der Löschmarkierung. Dieser Feldname wird in jeder Tabelle verwendet.

cBaseDataAccess.cDeletedFilter (vfxctrl.vcx) – Hier wird intern der Filterausdruck gespeichert, der im *BeforeCursorFill()* Ereignis der Where Klausel hinzugefügt wird um gelöschte Datensätze auszuschließen.

cBaseDataAccess.IDelFieldSet (vfxctrl.vcx) – Hier wird intern gespeichert, ob bei der Aktualisierung ein Löschvorgang durchgeführt wurde.

Methoden

cDataFormVFXBase.SetRecordDeleted() (vfxformbase.vcx) – Mit dieser Methode wird der Löschvorgang durchgeführt. Diese Methode wird statt des *Delete* Befehls aufgerufen. Abhängig vom Wert der Eigenschaft *cDel_fld* markiert diese Methode einen Datensatz als gelöscht oder löscht den Datensatz.

Wenn ein Datensatz durch Setzen der Löschmarkierung gelöscht wird, führt der Cursoradapter keinen *Delete* Befehl, sondern einen *Update* Befehl aus, weil der Wert des Feldes mit der Löschmarkierung aktualisiert werden muss.

Wenn ein Datensatz durch Setzen der Löschmarkierung gelöscht wird, wird der Wert der Eigenschaft *IDelField-Set* des Cursoradapters im Ereignis *BeforeUpdate* auf *.T.* eingestellt und im Ereignis *AfterUpdate* wird der Datensatz aus dem Cursor gelöscht und der Löschstaus des Datensatzes wird mit *SETFLDSTATE(0,1)* zurückgesetzt, um nicht ein neues Aktualisierungsereignis auszulösen.

Die Einstellung von *SET DELETED* wird beachtet, wenn ein Cursoradapter mit Daten gefüllt wird.

Wenn *SET DELETED OFF* eingestellt ist, werden alle Datensätze geholt. Wenn *SET DELETED ON* eingestellt ist, werden nur Datensätze geholt, deren Löschmarkierung den Wert 0 hat.

Schreibschutzmarkierung

Durch Setzen einer Schreibschutzmarkierung kann ein Datensatz vor der Bearbeitung geschützt werden. Die Eigenschaft *goProgram.cRdn_Fld* enthält den Namen des Feldes, das die Schreibschutzmarkierung enthält. Der Schreibschutz funktioniert nur dann, wenn eine Tabelle dieses Feld enthält und der Wert der Eigenschaft *goProgram.cRdn_Fld* auf *.T.* eingestellt ist.

Die neue Methode *cDataFormVFXBase.GetReadOnlyStatus()*, wird von der Methode *OnRecordMoveRefresh()* in der Klasse *cDataFormVFXBase* aufgerufen. Diese Methode prüft, ob das Feld mit der Schreibschutzmarkierung vorhanden ist und ob ein Schreibschutz besteht. Wenn ja, werden die Eigenschaften *ICanEdit* und *ICanDelete* des Formulars auf *.F.* gesetzt.

Berechtigungen auf Datensatzebene

In VFX 10.0 Anwendungen kann eine Berechtigung für bestimmte Benutzer für jeden Datensatz eingestellt werden. In der Tabelle mit den Berechtigungen kann der Primärschlüssel aus der Benutzertabelle verwendet werden.

Die Tabelle für Berechtigungen unterstützt jetzt die Felder für den Löschstaus (*goProgram.cDel_Fld*), den Namen des Benutzers, der den Datensatz neu angelegt hat (*goProgram.cIns_Usr*), das Datum, an dem der Datensatz neu angelegt wurde (*goProgram.cIns_Date*), den Namen des Benutzers, der den Datensatz zuletzt bearbeitet hat (*goProgram.cEdt_Usr*) sowie das Datum, an dem der Datensatz zuletzt bearbeitet wurde (*goProgram.cEdt_Date*). Wenn ein oder mehrere dieser Felder in der Tabelle für Berechtigungen existieren, werden diese Felder genauso wie in anderen Tabellen behandelt und von VFX automatisch mit den entsprechenden Werten gefüllt.

Wenn ein neuer Datensatz der Tabelle für Berechtigungen hinzugefügt wird, der aber bereits mit gelöschtem Löschstaus existiert, wird der Löschstaus des existierenden Datensatzes zurückgesetzt und es wird kein neuer Datensatz angelegt.

Der Code zur Implementierung dieser Funktionalität befindet sich in der Klasse *cDataFormVFXBase* in der Methode *CallSecurityRightsDialog*.

Neue Eigenschaften des Anwendungsobjekts goProgram (Klasse cFoxApp)

Installation eines Postscript Druckertreibers

Mit einer Eigenschaft von *goProgram* kann eingestellt werden, ob ein Postscript Druckertreiber installiert werden soll, auch wenn bereits ein anderer Postscript Druckertreiber installiert ist.

AlwaysInstallPSPrinter – (Standardwert .F.) Diese Eigenschaft steuert die Installation eines Postscript Druckertreibers. Ein Postscript Druckertreiber wird für die Erstellung von PDF Dateien benötigt. Wenn der Wert dieser Eigenschaft auf .T. gestellt ist, wird der Druckertreiber auf jeden Fall installiert, auch wenn bereits ein anderer Postscript Druckertreiber vorhanden ist. Der Name des zu installierenden Postscript Druckertreibers steht in der Eigenschaft *goProgram.PSPrinterToInstall*.

Datenverwaltung

nConfigVfxFormat – In dieser Eigenschaft wird das Format angegeben, in dem die Datei Config.vfx gespeichert wird. 0 - XML (Standardwert), 1 – CSV.

nShowRetrieveMsg – Mit dieser Eigenschaft kann eingestellt werden, ob die Meldung „Daten holen...“ während der Anforderung von Daten eines Cursoradapters angezeigt wird. 0 – (Standardwert) – Es gilt die Einstellung des Cursoradapters, 1 – Meldung immer anzeigen, 2 – Meldung nie anzeigen.

cDel_Fld – Diese Eigenschaft enthält den Namen des Feldes, in dem der Löschstaus eines Datensatzes gespeichert wird. Dieser Feldname wird in allen Tabellen verwendet. Wenn dieses Feld in einer Tabelle nicht existiert, wird der Datensatz physikalisch gelöscht.

cRdn_Fld – Diese Eigenschaft enthält den Namen des Feldes, in dem der Schreibschutzstatus eines Datensatzes gespeichert wird. Dieser Feldname wird in allen Tabellen verwendet.

IFillEditDate – Wenn der Wert dieser Eigenschaft auf .T. eingestellt ist, wird bei neu angelegten Datensätzen der Wert des Feldes *edt_date* auf den gleichen Wert gesetzt, den auch der Wert des Feldes *ins_date* hat.

UseGUIDKeys – Wenn der Wert dieser Eigenschaft auf .T. eingestellt ist werden GUID Felder anstelle von Integer Autoinc Feldern verwendet. In den Formulklassen werden nach dem Einfügen neuer Datensätze GUIDs erzeugt.

Produktaktivierung über das HTTP Protokoll

Die Registrierung von VFX Anwendungen ist jetzt wahlweise über das HTTP Protokoll möglich. Der bisher verwendete Web Service steht weiterhin zur Verfügung. Um in einer Anwendung das HTTP Protokoll zu verwenden, muss der Wert der Eigenschaft *cVFXActivation.nRegWay* (zu finden in appl.vcx) auf 13 gestellt werden. Die URL des Registrierungsservers muss in der Eigenschaft *cVFXActivation.cHTTPRegisterURL* (appl.vcx) angegeben werden. Diese Einstellung kann auch im VFX – Application Builder gemacht werden. Der Prozeß der Registrierung über HTTP ist vergleichbar mit der Registrierung über den Web Service. Für die Registrierung über das HTTP Protokoll wird die Methode *RegisterCustomerViaHTTP()* in der Klasse *cConnectWebService* verwendet. Für die Registrierung über den Web Service wird die Methode *RegisterCustomer()* verwendet.

Die Registrierung über das HTTP Protokoll verwendet Windows API Funktionen aus der

wininet.dll um die Verbindung zum Registrierungsserver herzustellen und die Registrierungsdaten in einem Textformat zu übertragen. Um die Registrierungsdaten in das erforderliche Format zu konvertieren werden die Funktionen *CSVStringToCursor* und *CursorToCSVString* verwendet.

Der Rückgabewert von der HTTP Registrierung wird in der Eigenschaft *cConnectWebService.cXML gespeichert*. Wenn bei der Registrierung ein Fehler auftritt, wird die Fehlermeldung in der Eigenschaft *cConnectWebService.cLastErrorText* gespeichert.

Die Klasse cVFXActivate (vfxappl.vcx)

nProductActivationBehavior – 1 – Das Format der Aktivierungsschlüssel ist kompatibel zu VFX 9.5, 2 – Das Format der Aktivierungsschlüssel ist kompatibel zu Microsoft Produkten.

cHTTPRegisterURLServerName – In dieser Eigenschaft steht der Name des Servers für die HTTP Aktivierung der Anwendung.

cHTTPRegisterURLObjectName – In dieser Eigenschaft steht der Name der ASP Seite für die HTTP Aktivierung der Anwendung.

cRegisterFormName – Diese Eigenschaft enthält den Namen des Formulars, das für die Registrierung verwendet wird. Der Standardwert ist *vfxRegister*.

Beispiel:

Die Webseite für die HTTP Registrierung hat die URL:
`http://www.vfxserver.org/vfxtestregistration/Register.asp`

In den Eigenschaften muss dafür eingestellt werden:
`cHTTPRegisterURLServerName = "www.vfxserver.org"`
`cHTTPRegisterURLObjectName = „vfxtestregistration/Register.asp“`

Die Klasse **cConnectWebService (vfxappl.vcx)**

Methoden

RegisterCustomerViaHTTP – Diese Methode stellt die Verbindung zum Registrierungsserver her und sendet die Kundendaten über das http Protokoll. Wenn als Rückgabewert ein gültiger Aktivierungsschlüssel geliefert wird, wird die Anwendung automatisch aktiviert.

Die Klasse **cRegistration (regservice.vcx im RegistrationWebService Projekt)**

Der Web Service für die Registrierung wurde so geändert, dass das empfangene Datenformat automatisch erkannt wird.

Eigenschaften

IDataInXMLFormat – Nur intern verwendet. Der Wert dieser Eigenschaft ist .T. wenn XML Daten empfangen wurden.

Methoden

CursorToString() – Konvertiert die übergebenen Daten in eine Zeichenkette. Das verwendete Format wird entsprechend der Einstellung in der Eigenschaft *IDataInXMLFormat* verwendet.

CursorToCSV() – Konvertiert den übergebenen Cursor in eine Zeichenkette im CSV Format mit einer zusätzlichen Kopfzeile mit Strukturinformationen.

CursorToXML() – Konvertiert den übergebenen Cursor in eine Zeichenkette im XML Format.

CSVToCursor() – Erstellt einen Cursor aus einer im CSV Format übergebenen Zeichenkette.

XMLToCursor() – Erstellt einen Cursor aus einer im XML Format übergebenen Zeichenkette.

Wichtig: Die DLL für den Registrierungs Web Service muss als Multi-threaded COM Server erstellt werden.

Vorbereiten einer Anwendung für die Produktaktivierung

Einstellungen im VFX – Application Builder

Im VFX – Application Builder muss zunächst die Produktaktivierung eingeschaltet werden. Dies geschieht mit der Checkbox „Enable Product Activation“. Die Wert der Eigenschaft *cFoxAppl.IUseActivation* kann wahlweise im Klassen-Designer auch manuell auf .T. eingestellt werden.

Der Aktivierungsschlüssel wird in einer Datei gespeichert. Der Name dieser Datei kann im VFX – Application Builder unter „Store activation data to“ eingetragen werden. Manuell kann der Wert der Eigenschaft *cVFXActivation.cStoreActivationData* eingestellt werden. Der Standardwert ist *VFX.ini*. Zusätzlich kann eingestellt werden, ob die Dateien mit den Informationen über die Produktaktivierung auf dem Kundenrechner versteckt werden sollen. Diese Einstellung kann in der Eigenschaft *cVFXActivation.IHideRegistrationFiles* gemacht werden. Der Wert kann auch im VFX – Application Builder im Kontrollkästchen *Hide registration files* gemacht werden.

Es besteht die Möglichkeit vom Web Service oder über die HTTP Aktivierung Aktivierungsschlüssel automatisch erstellen zu lassen, die zeitlich befristet sind. So kann man interessierten Kunden die Möglichkeit geben, die Anwendung in einem festgelegten Umfang testen zu können. Hier muss im VFX – Application Builder „Activation key validity in days“ eingestellt werden. Der Standardwert ist 30 Tage.

Bei „Activation key type“ kann das zu bisherigen VFX Versionen kompatible Format des Aktivierungsschlüssels gewählt werden. Diese Aktivierungsschlüssel können relativ lang werden. Wahlweise kann auch ein kürzeres Format für den Aktivierungsschlüssel verwendet werden. Bei diesem Format ist der Aktivierungsschlüssel immer genau 25 Zeichen lang. Jeweils fünf Stellen sind durch einen Bindestrich getrennt. Anwender kennen dieses Format von Aktivierungsschlüsseln von verschiedenen Microsoft Produkten. Im VFX – Application Builder wird „Activation key type“ eingestellt. Manuell kann der Wert der Eigenschaft `cVFXActivation.nProductActivationBehavior` auf 1 für lange Aktivierungsschlüssel oder 2 für kurze Aktivierungsschlüssel eingestellt werden.

Die Checkbox „Time limited activation key“ muss markiert werden, wenn befristet gültige Aktivierungsschlüssel erstellt werden sollen. Diese Checkbox ist nur dann enabled, wenn kurze Aktivierungsschlüssel verwendet werden. Manuell kann der Wert der Eigenschaft `cVFXActivation.IUseTimeLimitedActivationKey` eingestellt werden.

Im Eingabefeld „Start day of activation key“ kann das Startdatum eingegeben werden, ab dem befristet gültige Aktivierungsschlüssel erstellt werden können. Der Standardwert ist der 01.01.2007. Manuell kann der Wert der Eigenschaft `cVFXActivation.dStartActivationDate` eingestellt werden.

Wenn mehrere Hardware Parameter zur Identifizierung des Kundenrechners verwendet werden, kann eine Toleranz bei der Überprüfung der Parameter eingestellt werden. Ein Aktivierungsschlüssel bleibt dann gültig, wenn der Kunde Änderungen an der Hardware vornimmt, solange die Anzahl der Änderungen die erlaubte Toleranz nicht überschreitet. Die Anzahl der erlaubten Hardware-Änderungen ist in der Eigenschaft `cVFXActivation.nHardwareParametersTolerance` eingestellt. Diese Einstellung kann auch im VFX – Application Builder unter *Number of changes accepted when using hardware parameters tolerance* eingestellt werden.

Bei der Aktivierung der Anwendung werden die Hardware Parameter in einer Datei gespeichert. Der Name dieser Datei kann in der Eigenschaft `cVFXActivation.cStoreHardwareParameters` oder im VFX – Application builder unter *Hardware parameters file* eingestellt werden. Der Standardwert ist `vfx.hrd`.

Solange eine Anwendung nicht aktiviert ist, wird beim Start der Anwendung ein Dialog zur Registrierung angezeigt. Der Name dieses Dialogs kann in der Eigenschaft

`cVFXActivation.cRegisterFormName` eingestellt werden. Im VFX – Application builder kann diese Einstellung unter *Name of the register form* gemacht werden. Der Standardwert dieser Eigenschaft ist `vfxactivationwizard` um den neuen VFX – Aktivierungsassistenten zu starten. Um den Registrierungsdialog aus VFX 9.5 anzuzeigen, muss der Wert dieser Eigenschaft auf `vfxRegister` eingestellt werden.

Vorbereitung des Servers

Zunächst muss auf dem Server die Laufzeitumgebung von VFP 9 installiert werden.

Dateien

Hier eine Auflistung, der auf dem Server für die Produktaktivierung erforderlichen Dateien:

- registrationwebservice.dll
- registervfx.dll
- vfxdownload.dll (alle DLLs müssen registriert werden)
- VFX.fl
- RegisterVFX.asp
- DownloadVFX.asp
- Test.asp (optional)
- vfxgenactkey.APP
- Config.vfx
- Log.txt (mit Schreibrechten, wenn die Verarbeitung protokolliert werden soll)

Weitere erforderliche Einstellungen

Für die Aktualisierung der Endanwendung muss der ASP Benutzer Leserechte für die zu versendende Installationsdatei haben.

Um die Größe des Puffers zu setzen, muss in der Datei AspBufferingLimit.vbs der Ordnername sowie die Puffergröße eingestellt werden. Anschließend muss diese Datei ausgeführt werden.

In der Datei AspBufferingLimit.vbs befindet sich die Zeile:

```
dirObj.AspBufferingLimit = 20971520
```

Diese Einstellung entspricht einer Puffergröße von 20 MB. Die Puffergröße muss entsprechend der Größe der Installationsdatei angepasst werden.

In dieser Datei befindet sich auch eine Zeile, die den Pfad zum Webordner im Objekt dirObj speichert. Wenn der Webordner ein Ordner unter Wwwroot ist, muss der Pfad in dieser Zeile eingetragen werden.

```
set dirObj =  
providerObj.get ("IISWebDirectorySetting='W3SVC/1/ROOT/VFX' ")
```

Wenn der Webordner ein virtueller Ordner ist, muss die obige Zeile kommentiert werden und die folgende Zeile muss unkommentiert werden.

```
'set dirObj =  
providerObj.get ("IISWebVirtualDirSetting='W3SVC/1/ROOT/VFX' ")
```

Hier muss der Pfad der Webanwendung eingetragen werden.

Anschließend muss die Datei AspBufferingLimit.vbs ausgeführt werden.

Einstellungen im Internet Information Server 7

Die ASP Seite für die HTTP Registrierung kann auf mit dem Internet Information Server 6 auf Windows 2000, Windows XP oder Windows Server 2003 ohne besondere Einstellungen ausgeführt werden.

Für den Internet Information Server 7, der standardmäßig auf Windows Vista installiert wird bzw. auf älteren Windows Versionen nachträglich installiert werden kann, müssen einige Einstellungen vorgenommen werden.

Wenn der IIS auf einer 64 bit Windows Version läuft, muss der IIS so eingestellt werden, dass er im 32 bit Modus läuft. Nur so können VFP Objekte instanziiert werden. Das geschieht mit dem Befehl:

```
cscript %SYSTEMDRIVE%\inetpub\adminscripts\adsutil.vbs SET  
W3SVC/AppPools/Enable32bitAppOnWin64 1
```

Als Rückmeldung sollte erscheinen:

```
Enable32bitAppOnWin64      : (BOOLEAN) Wahr
```

Danach muss IIS neu gestartet werden.

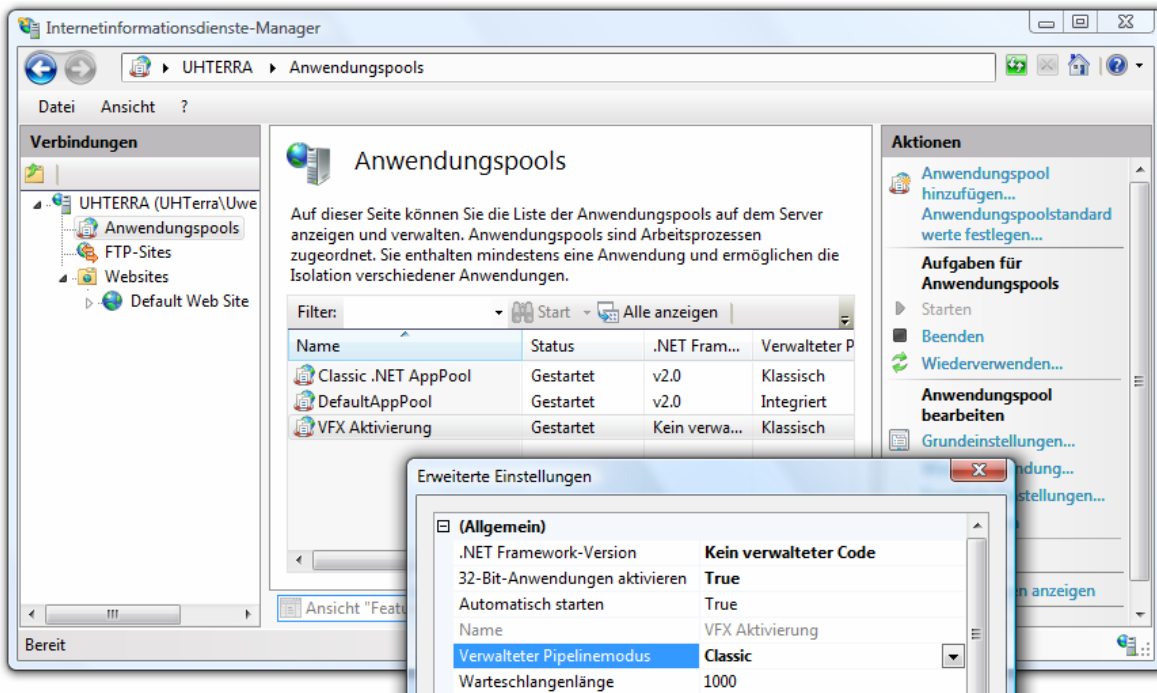
Im Internetinformationsdienste-Manager muss bei Anwendungspools ein neuer Anwendungspool hinzugefügt werden. Der Name kann beliebig gewählt werden, zum Beispiel VFX-Aktivierung. Folgende Einstellungen sind für den neuen Anwendungspool zu machen:

.NET Framework Version	Kein verwalteter Code
Verwalteter Pipeline Modus	Klassisch

Anschließend ist in den erweiterten Einstellungen des neu erstellten Anwendungspools

```
32-bit-Anwendungen aktivieren    True
```

einzustellen.

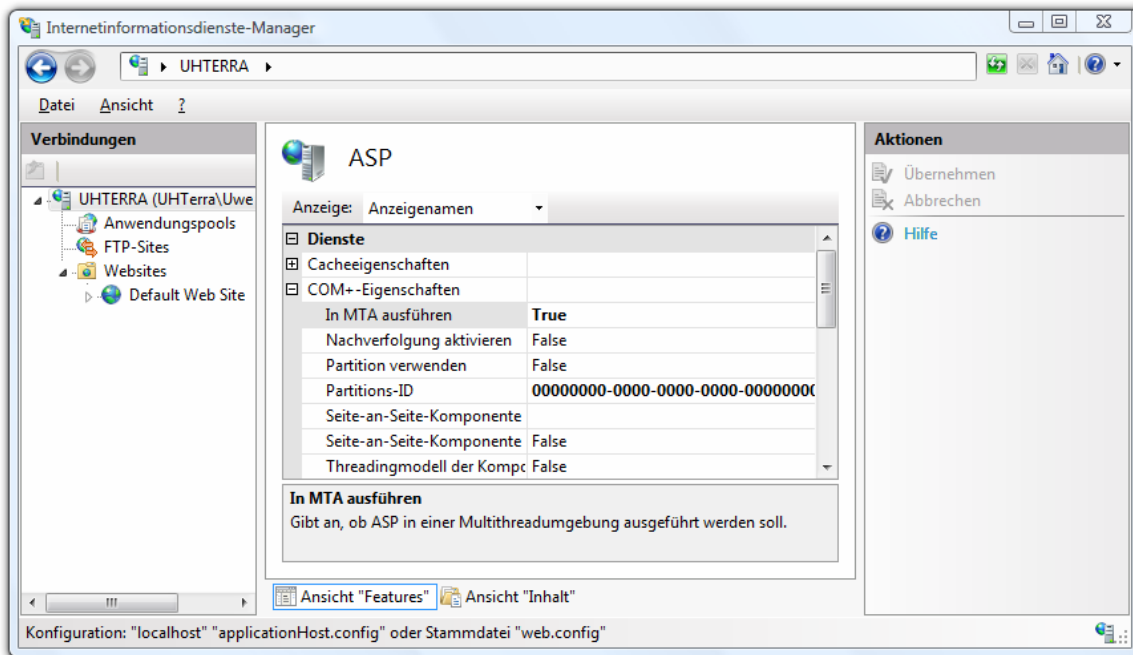


Unter Websites ist unter der verwendeten Website, in der Regel „Default Web Site“, eine Anwendung hinzuzufügen. Dafür ist der Anwendungspool auf den im letzten Schritt erstellten Anwendungspool einzustellen, zum Beispiel VFX-Aktivierung.

Auf der Seite ASP im Internetinformationsdienste-Manager muss bei COM Eigenschaften

In MTA ausführen True

eingestellt werden.



VFX – Aktivierungsassistent

Um die Registrierung für Endanwender zu vereinfachen gibt es den Aktivierungsassistenten. Der Aktivierungsassistent befindet sich in der Klasse *cActivationWizardVfxBase* in der Klassenbibliothek *VfxFormBase*. Eine 1:1 Ableitung mit dem Namen *cActivationWizard* befindet sich in der Klassenbibliothek *VfxForm*. Der Aktivierungsassistent basiert auf der Klasse *cWizard*. Für den Aktivierungsassistenten gibt es das Formular *VfxActivationWizard*.

Der Klasse *cVfxActivate* aus der Klassenbibliothek *VfxAppl.vcx*, wurde die neue Eigenschaft *cRegisterFormName* hinzugefügt. Diese Eigenschaft enthält den Namen des Formulars, in dem die Registrierungsdaten eingegeben werden. Der Standardwert ist *vfxRegister*. Die Anwendung verwendet den neuen Aktivierungsassistenten, wenn der Wert dieser Eigenschaft auf *VfxActivationWizard* eingestellt ist. Es ist auch möglich ein eigenes Formular zur Eingabe der Registrierungsdaten zu erstellen und den Namen des Formulars in dieser Eigenschaft einzutragen.

Die Klasse **cRegisterProcessing**

Zur einfachen Handhabung vom Registrierungsdialog und dem Aktivierungsassistenten ist die wesentliche Funktionalität der Registrierung in der Klasse *cRegisterProcessing* in der Klassenbibliothek *VfxAppl.vcx* gekapselt.

Eigenschaften

cListNotSetProperties – Nur intern verwendet. In dieser Eigenschaft wird eine Liste der Eigenschaften ohne Wert verwaltet. Die Liste der fehlenden Eigenschaften wird im Aktivierungsassistenten angezeigt.

lRegEmailNotAvailable – Nur intern verwendet. Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, wird eine Meldung angezeigt, weil keine E-Mailadresse für die Registrierung angegeben ist oder die E-Mail Option in der Klasse *cActivationWizardVfxBase* disabled ist.

lRegPropertiesNotSet – Der Wert dieser Eigenschaft wird intern auf *.T.* gesetzt, wenn die zur Registrierung erforderlichen Eigenschaften nicht gesetzt sind.

Methoden

CheckRegistrationProperties – Überprüft, ob alle zur Registrierung erforderlichen Eigenschaften eingestellt sind.

CheckRequiredFields – Überprüfung der Pflichtfelder auf Eingaben. Pflichtfelder sind im Formular durch ein * markiert.

CreateRegInfoTable – Erstellung des Cursors *RegInfo* mit Registrierungsdaten.

ErrorMessage – Anzeige einer Fehlermeldung, wenn das Valid Ereignis *.F.* zurückgegeben hat.

ErrorMessage_SendPassword – Anzeige einer Fehlermeldung, wenn der Versand des Kennworts fehlgeschlagen ist.

GetApplicationVersion – Rückgabe der Versionsnummer der Anwendung.

GetCaption – Rückgabe der Bezeichnung eines Labels.

GetEmailText – Zusammenstellen des Textes für die Registrierungs-E-Mail.

PrepareCustomerData – Vorbereitung der Kundendaten.

PrepareRequestReplacementKeyData – Vorbereitung des zu sendenden Ersatzschlüssels.

PrepareSendPasswordData – Vorbereitung des zu sendenden Kennworts.

ProcessActKey – Überprüfung des empfangenen Aktivierungsschlüssels.

RegisterOnlineErrorMsg – Anzeige einer Fehlermeldung, wenn die Registrierung fehlschlägt.

RunWithoutRegistration – Diese Methode wird ausgeführt, wenn der Anwender *Ohne Registrierung testen* wählt.

SaveToFile – Speichern der Registrierungsinformationen in einer Datei. Der Name der Datei kann in der Eigenschaft *cParamfile* eingestellt werden.

SendEmail – Versand einer E-Mail mit den Registrierungsdaten.

SendPasswordViaEmail – Aufruf der Methode des Web Service zur Anforderung des Kennworts.

SendRegInfo – Versand der Registrierungsdaten über den Web Service oder über HTTP.

TryAutomaticRegistration – Automatische Aktivierung der Anwendung, wenn im Anwendungsordner die *XAK* Datei mit dem Aktivierungsschlüssel gefunden wird.

ValidActKey – Überprüfung eines Aktivierungsschlüssels.

ValidateData – Überprüfung der eingegebenen Registrierungsdaten. Diese Methode wird von der Methode *CheckRequiredFields()* aufgerufen.

Die Klasse **cActivationWizardVfxBase**

Eigenschaften

cParamFile – Name der Datei, in der die Registrierungsdaten gespeichert werden.

cRegEmail – An die hier angegebene E-Mailadresse werden die E-Mails mit Registrierungsdaten gesendet.

cRegistrationResultCaption – Nur intern verwendet. Überschrift der Ergebnismeldung der Aktivierung.

cRegistrationResultMessage – Nur intern verwendet. Meldungstext der Ergebnismeldung der Aktivierung.

cServiceMethodName – Name der aufzurufenden Methode des Web Service.

cServiceName – Name des Web Service Dienstes.

cWSActivationKeyFieldName – Name des Feldes mit dem Aktivierungsschlüssel.

cWSDL –URL zur WSDL Datei des Web Service.

cWSResultAddInfoFieldName – Name des Feldes in der XML oder CSV Zeichenkette mit zusätzlichen Informationen. Dieser Feldname wird vom Web Service oder von der Registrierungs-Website zurückgegeben.

cWSResultStatusFieldName – Name des Feldes in der XML oder CSV Zeichenkette mit dem Status. Dieser Feldname wird vom Web Service oder von der Registrierungs-Website zurückgegeben.

INIFilename – Nur intern verwendet. Hier steht der Pfad- und Dateiname zu der Ini Datei mit den Aktivierungsinformationen.

nRegWay – Mit dem Wert dieser Eigenschaft wird eingestellt, auf welchem Weg die Registrierungsdaten an den Entwickler gesendet werden.

oActivation – Diese Eigenschaft enthält eine Referenz auf das Objekt *goActivation*, weil dieses Objekt während der Initialisierung dieses Objekts noch nicht global zur Verfügung steht.

Methoden

GetMessageText – Rückgabe eines Textes in der aktuellen Sprache.

LoadLicenseAgreement – Laden des Textes des Lizenzvertrages.

LoadReasonCombo – Laden der lokalisierten Texte mit Gründen für Ersatzschlüssel.

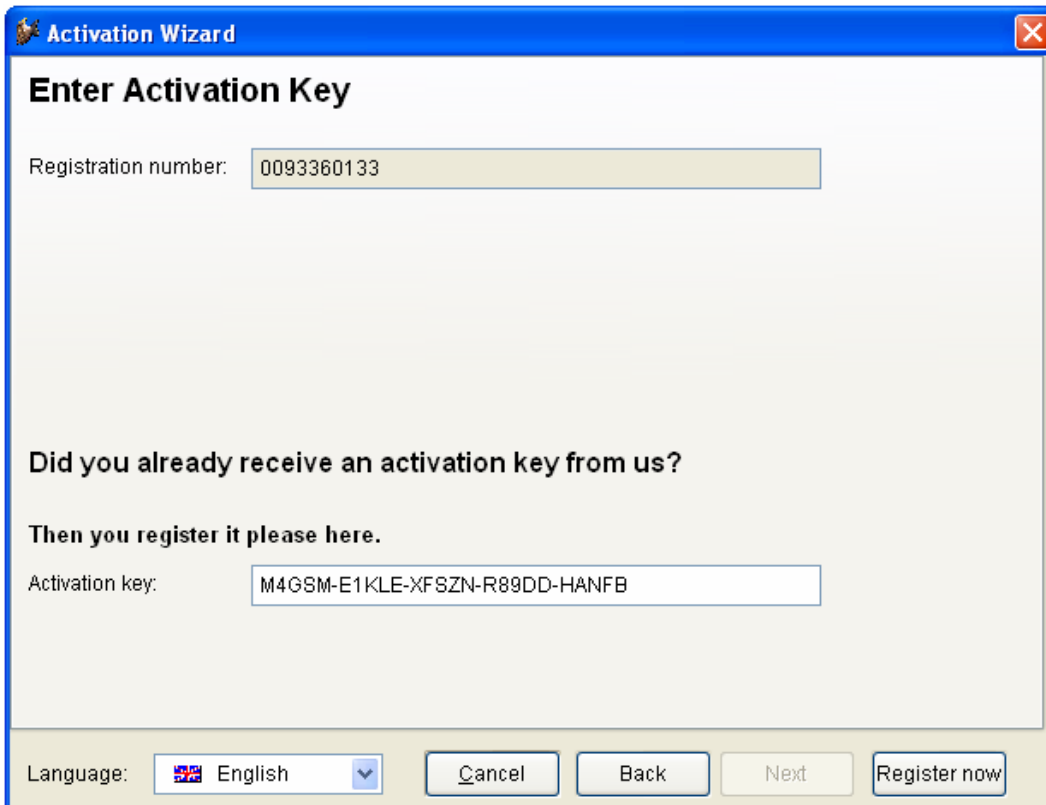
OnSuccessfulActivation – Diese Methode wird nach der erfolgreichen Aktivierung ausgeführt.

Bedienung

Auf der ersten Seite des Aktivierungsassistenten kann ein vorhandener Aktivierungsschlüssel direkt eingegeben werden.

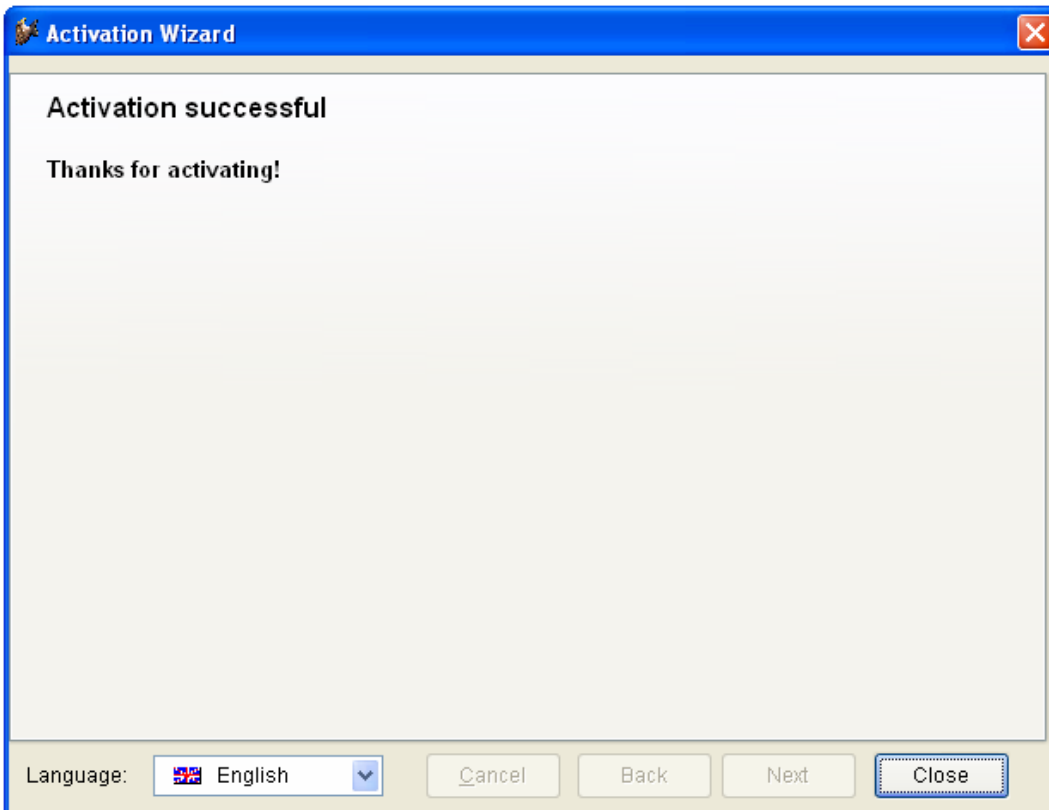


Wenn der Benutzer auf die Schaltfläche „Aktivierungsschlüssel eingeben“ klickt, erscheint die folgende Seite.



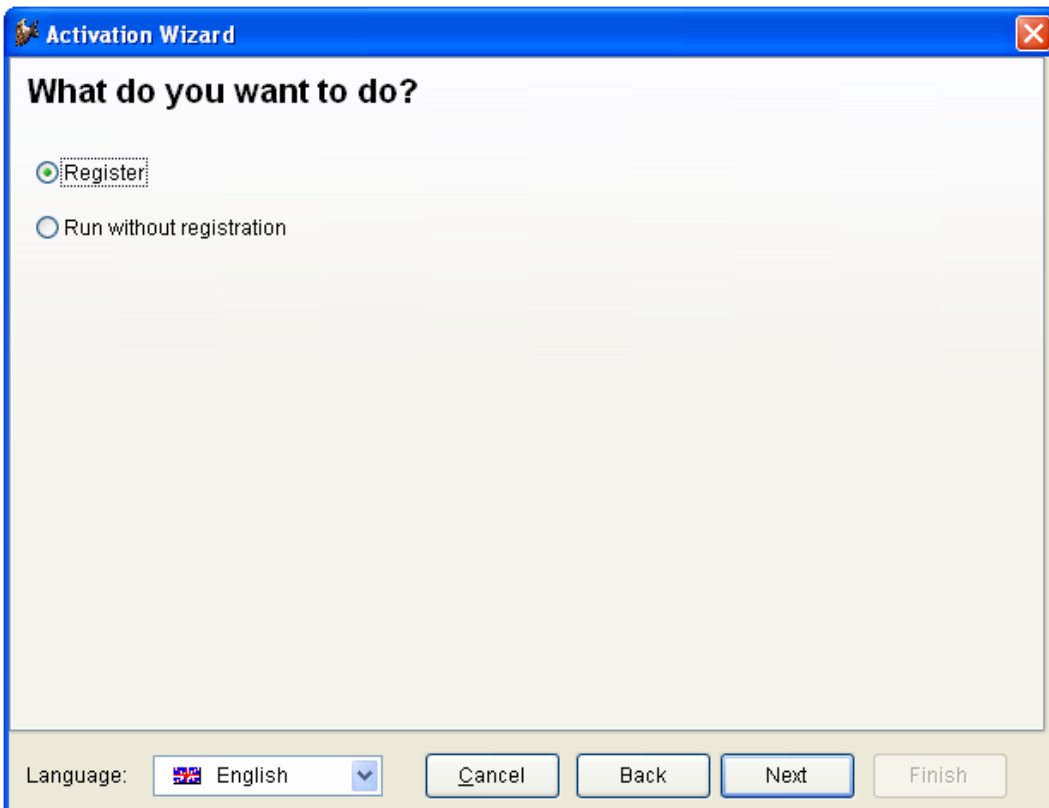
Wenn der Benutzer nach der Eingabe des Aktivierungsschlüssels auf die Schaltfläche „Jetzt registrieren“ klickt, wird der Aktivierungsschlüssel auf Gültigkeit überprüft.

Wenn der Aktivierungsschlüssel gültig ist, wird die Anwendung aktiviert und die letzte Seite des Aktivierungsassistenten wird angezeigt.



Dies ist die letzte Seite des Aktivierungsassistenten. Hier werden Meldungen über den Verlauf der Aktivierung angezeigt. Durch einen Klick auf die Schaltfläche „Schließen“ wird der Aktivierungsassistent geschlossen.

Wenn auf der ersten Seite des Aktivierungsassistenten die Schaltfläche „Weiter“ betätigt wird, erscheint die zweite Seite, auf der der Benutzer den Weg der Registrierung wählen kann.



In diesem Schritt kann der Benutzer wählen, ob er die Anwendung registrieren will oder die Ausführung ohne Registrierung fortsetzen will. Wenn der Benutzer die Anwendung nicht registrieren will, wird die der Aktivierungsassistent beendet und die Ausführung der Anwendung wird ohne Aktivierung fortgesetzt.

Wenn der Benutzer die Anwendung registrieren will, geht es mit dem nächsten Dialogschritt weiter.

Activation Wizard

End-user license agreement

VIEW THE END-USER LICENSE AGREEMENT

I hereby accept this license agreement and confirm that I have received, read and understood the terms and provisions therein

I do not agree with the license agreement.

Language: English

Cancel Back Next Finish

Hier wird dem Anwender der Lizenzvertrag angezeigt. Die Ausführung des Aktivierungsassistenten kann nicht fortgesetzt werden, solange der Anwender den Lizenzvertrag nicht akzeptiert.

Activation Wizard

Please fill the following data:

User account information

E-mail: *

Password: *

Confirmation: *

Customer information

First name: *

Last name: *

Company:

Street:

Zip Code:

City:

State:

Country: Bulgaria

e-mail notification

Phone:

Fax:

Tax ID number:

Bank account information

Bank name:

Bank code:

Bank account:

* required fields

Version #:

Your Registration Key is:

Language: English

Cancel Back Next Finish

In diesem Schritt gibt der Anwender seine persönlichen Daten ein. Die Ausführung des Aktivierungsassistenten kann nur dann ausgeführt werden, wenn in allen Pflichtfeldern Eingaben gemacht sind. Es muss eine gültige E-Mailadresse eingegeben werden. Das Kennwort muss mindestens fünf Zeichen lang sein. Das Kennwort und die Bestätigung des Kennworts müssen identisch sein.

Der nächste Dialogschritt hängt davon ab, wie die Anwendung die Anwendung aktiviert wird. Wenn die Aktivierung so eingestellt ist, dass nicht online aktiviert ist, erhält der Anwender eine Meldung die erklärt, wie die Aktivierung durchgeführt wird. Wenn online mittels des Web Service oder der http Aktivierung aktiviert wird, erscheint der folgende Dialogschritt:

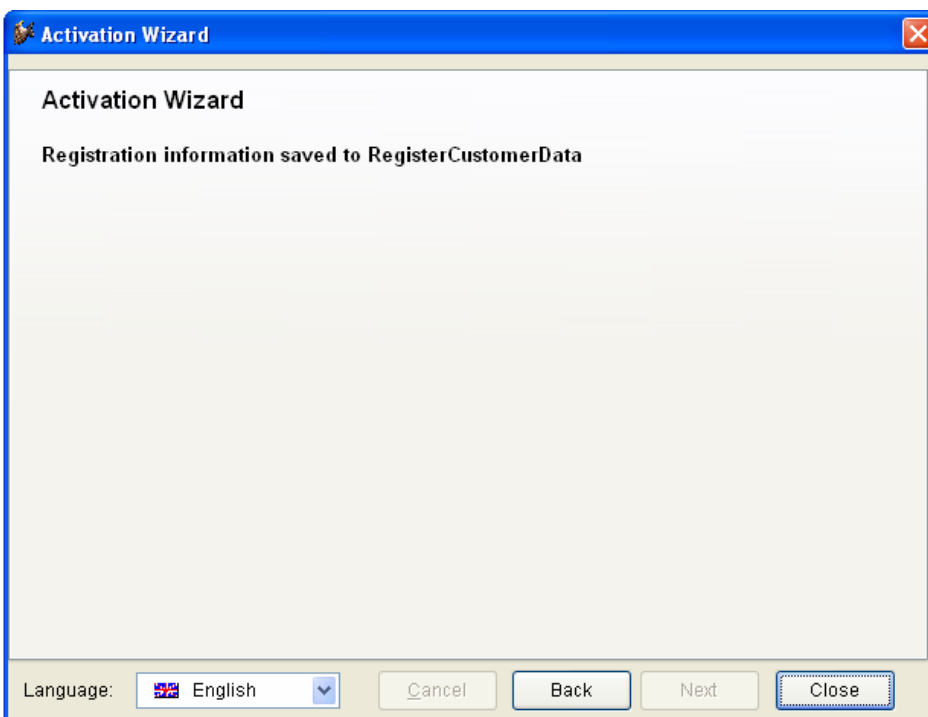


Die Option *Registrierungs-E-Mail* senden erscheint nur, wenn die Anwendung so konfiguriert ist, dass eine Registrierungs-E-Mail gesendet werden kann. Die Eigenschaft *cRegEmail* muss hierfür eine E-Mailadresse enthalten.

Die Registrierungsmethode auf 11 (die Registrierungsdaten werden in einer Datei gespeichert) oder 12 (die Registrierungsdaten werden per E-Mail gesendet) gesetzt ist, wird in diesem Schritt nur eine Meldung angezeigt.

Wenn der Anwender auf die Schaltfläche *Beenden* klickt, beginnt der Registrierungsprozess.

Die letzte Seite des Aktivierungsassistenten zeigt eine abschließende Meldung über den Registrierungsprozess. Wenn die Registrierung erstmalig stattfindet, erscheint die Meldung:



Durch einen Klick auf die Schaltfläche *Schließen* wird der Aktivierungsassistent beendet.

Der Aktivierungsassistent ist lokalisiert und wenn die Anwendung mit Lokalisierung zur Laufzeit arbeitet, kann die Sprache zu jeder Zeit gewechselt werden. Der Aktivierungsassistent startet in der Sprache entsprechend den Regionaleinstellungen von Windows.

Zu jeder Zeit kann der Anwender die Ausführung des Aktivierungsassistenten durch einen Klick auf die Schaltfläche *Abbrechen* beenden. In diesem Fall wird der Aktivierungsassistent beendet und die Anwendung läuft ohne Aktivierung.

Die Klasse **cFormBase**

Methoden

SetAlwaysOnTopOnForms(tlSet, tlIgnoreWindowType)

tlSet – Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, speichert diese Methode Referenzen alle Formulare, bei denen die Eigenschaft *AlwaysOnTop* den Wert *Wert.T.* hat, in dem eindimensionalen Array *thisform.aalwaysontop[]* und setzt die Eigenschaft *AlwaysOnTop* bei diesen Formularen anschließend auf *.F.* Eine Referenz des aktuellen Formulars wird nicht in dem Array gespeichert.

Wenn der Wert dieser Eigenschaft auf *.F.* eingestellt ist und das Array *thisform.aalwaysontop[]* nicht leer ist, weist diese Methode allen Formularen, deren Referenz im Array gespeichert ist, der Eigenschaft *AlwaysOnTop* den Wert *.T.* zu.

tlIgnoreWindowType – Wenn der Wert dieser Eigenschaft auf *.F.* eingestellt ist, wird diese Methode nur ausgeführt, wenn das Formular modal ist. Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, wird diese Methode in jedem Fall ausgeführt.

Diese Methode wird während der Initialisierung von Formularen mit dem Parameter *tlSet = .T.* aufgerufen sowie während des Release Ereignisses von Formularen mit dem Parameter *tlSet = .F.* Diese Methode wird auch vor und nach Anzeige der Seitenansicht von Berichten aufgerufen, um zu verhindern, dass Formulare, die immer im Vordergrund (zum Beispiel Toolbox) erscheinen, vor der Seitenansicht von Berichten erscheinen, wenn *Set Reportbehavior* 80 eingestellt ist.

Die Klasse **cDataFormVFXBase**

Eigenschaften

aAdditionalControlsToRequery – Dieses Array enthält Referenzen zu allen Objekten, die auf der Klasse *cComboPicklist* basieren. Von diesen Steuerelementen muss die *Requery* Methode ausgeführt werden, wenn der Satzzeiger bewegt wird und wenn der Wert der Eigenschaft *cComboPicklist.AddCurrentValueToList* auf *.T.* eingestellt ist.

oFooterControl – Diese Eigenschaft enthält eine Referenz auf den obersten *FooterBar* Container in einem Formular.

Methoden

SetRecordDeleted() – Diese Methode wird statt des *Delete* Befehls aufgerufen. In dieser Methode wird ein Datensatz als gelöscht markiert, indem die Löschemarkierung auf 1 gesetzt wird oder der Datensatz gelöscht wird. Das Verhalten hängt von der Einstellung der Löschemarkierung ab.

GetReadOnlyStatus() – Diese Methode wird von der Methode *OnRecordMoveRefresh()* aufgerufen. Diese Methode überprüft den Wert des *ReadOnly* Feldes, wenn es existiert. Wenn der aktuelle Datensatz als schreibgeschützt markiert ist, wird der Wert der Eigenschaften *ICanEdit* und *ICanDelete* des Formulars auf *.F.* eingestellt.

CallSecurityRightsDialog() – Anzeige des Dialogs mit Berechtigungen auf Datensatzebene.

Die Klasse cVfxActivate

Alle Computer-spezifischen Daten, ausgenommen Konstanten, werden in einer Parameterdatei gespeichert. Diese Daten werden bei der Toleranz für Parameter verwendet.

Eigenschaften

aInitialHardwareParametersValue() – In diesem Array werden die ursprünglichen Werte der Hardware-Parameter gespeichert.

cApplicationFileLocClassName() – Diese Eigenschaft enthält den Namen der verwendeten Klasse *ApplicationFileLoc*. Der Standardwert ist *cApplicationFileLoc*.

dEndValidityDate() – Diese Eigenschaft enthält das Gültigkeitsdatum des Aktivierungsschlüssels.

Methoden

CheckHardwareParametersValidity() – Diese Methode überprüft, ob die ursprünglichen Werte der Hardware-Parameter mit den aktuellen Werten übereinstimmen.

GetActKey(tcIniFile) – Diese Methode liest die Ini Datei und gibt den Aktivierungsschlüssel zurück, wenn ein Aktivierungsschlüssel vorhanden ist.

GetFileCreationDate(tcFile) – Diese Methode liefert das Erstellungsdatum der angegebenen Datei.

GetRegKeyValue(tcRegKey) – Diese Methode liefert den Wert des angegebenen Schlüssels aus der Windows-Registrierungsdatenbank.

LoadInitialHardwareParametersValue() – Diese Methode liest die der Hardware-Parameter und speichert die Werte in dem Array *aInitialHardwareParametersValue*.

Die Klasse cExport

Eigenschaften

cExportFields – Eine komma-separierte Liste von Feldnamen, die in den Export eingeschlossen werden sollen. Wenn der Wert dieser Eigenschaft nicht leer ist, werden in den Export nur die Felder einbezogen, die in dieser Eigenschaft angegeben sind, ausgenommen ist der Export in eine XML Datei.

Die Klasse cErrorReportDialog

Wenn die Eigenschaft *goActivation.nRegWay* den Wert 13 hat, werden Fehlerberichte über das HTTP Protokoll versendet. Die versendeten Daten werden im Textformat versendet.

Die Klasse cDocumentManagement

Drag and Drop

Von Ordern aus Outlook wird in der Dokumentenverwaltung die ID gespeichert. Dadurch ist es möglich die Namen der Ordner in Outlook zu ändern und die Verknüpfung aus der Dokumentenverwaltung bleibt erhalten.

Das Formular wechselt im *DragOver* Ereignis des Dokumentencontainers in den Bearbeitungsmodus. Dadurch ist es möglich mehrere Dateien in einem Drag & Drop Vorgang in die Dokumentenverwaltung einzufügen.

Die Klasse cGridMoverDialog

Im Auswahl-Grid und im Ziel-Grid ist inkrementelle Suche erlaubt.

Eigenschaften

oActiveColumn – Diese Eigenschaft enthält eine Referenz auf die Spalte, die den Fokus hat.

Methoden

onHeaderDbClick() – An diese Methode sind die *DbClick* Ereignisse aller Header gebunden.

onHeaderMouseDown() – An diese Methode sind die *MouseDown* Ereignisse aller Header gebunden.

onHeaderMouseEnter() – An diese Methode sind die *MouseEnter* Ereignisse aller Header gebunden.

onHeaderMouseUp() – An diese Methode sind die *MouseUp* Ereignisse aller Header gebunden.

onTextKeyPress() – An diese Methode sind die *KeyPress* Ereignisse aller Textboxen gebunden.

Eigenschaften für Endbenutzer

nGenerateOneToManyReport – Mit dieser Eigenschaft kann eingestellt werden, ob in OneToMany Formularen 1:n Berichte oder Berichte, die nur auf den Parent Daten basieren, generiert werden sollen. 0 – Formulareinstellung verwenden, 1- In allen Formularen .T., 2 – In allen Formularen .F.

nEnableChildInsert – Mit dieser Eigenschaft kann eingestellt werden, ob bei einem Klick in ein Childgrid ein neuer Datensatz angefügt werden soll. 0 – Formulareinstellung verwenden, 1 – In allen Formularen eingeschaltet, 2 – In allen Formularen ausgeschaltet.

nRecordMoveRefreshTimeout – Mit dieser Eigenschaft kann ein Intervall in Millisekunden eingestellt werden, das die Verzögerung einstellt, mit der das Formular nach dem Bewegen des Satzzeigers im Parent-Teil aktualisiert wird. Hierdurch wird die Ausführungsgeschwindigkeit, insbesondere bei der inkrementellen Suche im Parent Teil des Formulars erheblich verbessert. Eigener Code, der mit dieser Verzögerung ausgeführt werden soll, kann in der Formularmethode *OnRecordMoveRefresh()* gespeichert werden.

lShowNTLogonFieldInUserManagement – Mit dem Wert dieser Eigenschaft kann eingestellt werden, ob das Feld mit dem Windows Anmeldenamen in der Benutzerverwaltung angezeigt werden soll. Wenn der Wert .F. ist, wird das Feld nicht angezeigt. Wenn der Wert .T. ist, wird das Feld in der Benutzerverwaltung angezeigt und in diesem Fall wird das Feld auch bei der automatischen Anmeldung an der Anwendung benutzt.

oXPOpenCombo – Nur intern verwendet. In dieser Eigenschaft wird eine Objektreferenz auf das Objekt *cXPOpenCombo* gespeichert, wenn diese Combobox anstelle des Öffnen Dialogs verwendet wird.

Neue Methoden des Anwendungsobjekts goProgram (Klasse cFoxApp)

vfxInputBox(tcInputBoxPrompt, tcInputBoxCaption, tcDefaultValue, tnTimeout, tcTimeoutValue, tcCancel Value) – Diese Methode ruft die VFP Funktion *InputBox* auf und gibt die Rückgabewerte zurück. Der Vorteil dieser Methode ist, dass die Zugriffstasten für die Zwischenablage Strg+C, Strg+V und Strg+X funktionieren.

Die Parameter entsprechen denen der VFP Funktion *InputBox*.

tcInputBoxPrompt – Die in der Inputbox angezeigte Bezeichnung oberhalb des Eingabefeldes.

tcInputBoxCaption – Die in der Inputbox angezeigte Überschrift.

tcDefaultValue – Standardwert der Inputbox.

tnTimeout – Angabe eines Timeout in Millisekunden. Wenn hier 0 angegeben wird oder dieser Parameter nicht übergeben wird, bleibt die Inputbox angezeigt, bis der Benutzer auf eine der Schlatflächen OK oder Abbrechen klickt.

tcTimeoutValue – Standardrückgabewert bei Erreichen des Timeouts. Dieser Wert wird nie zurückgegeben, wenn für die Inputbox kein Timeout angegeben wurde.

tcCancelValue – Die hier angegebene Zeichenkette wird zurückgegeben, wenn der Benutzer auf abbrechen klickt oder die Taste Esc drückt.

Neue Eigenschaften der Klasse `cBaseDataAccess`

ShowRetrieveMsg – Mit dieser Eigenschaft kann eingestellt werden, ob die Meldung „Daten holen...“ während der Ausführung der Methoden *Cursorfill* oder *Cursorrefresh* von Cursoradaptern angezeigt wird. .T. – Anzeige der Meldung, .F. Keine Meldung anzeigen. Der Standardwert ist .F.

pickWhereClause – In dieser Eigenschaft wird eine Where-Klausel angegeben, die verwendet wird, wenn der Cursoradapter von der Klasse *cPickDialog* zur Anzeige einer Auswahlliste instanziiert wird. Die Where Klausel enthält normalerweise *This.vIDValue*, anstelle eines Parameters, der auf Daten verweist.

IDFieldFullName – Name des Arbeitsbereichs und Feldes, das als Parameter verwendet wird. Dieser Wert wird während der Initialisierung des Auswahllistendialogs in der Datensitzung des aufrufenden Formulars ausgeführt und in der Eigenschaft *vIDValue* gespeichert, um später verfügbar zu sein, wenn die Auswahlliste in Ihrer eigenen Datensitzung ausgeführt wird.

vIDValue – Nur intern verwendet. Diese Eigenschaft enthält den Wert der ID für die Where Klausel.

Beispiel:

```
cWhereClause = "field1 = ?tablename.field2"  
cIDFieldFullName = "tablename.field2"  
cPickWhereClause = "field1 = ?this.vIDValue"
```

Klasse `cXPOpenCombo (vfxappl.vcx)` – Combobox für den Start von Formularen

Durch dieses neue Element der Benutzeroberfläche ist es möglich eine Combobox in der Standardsymbolleiste anstelle des Öffnen-Dialogs zu verwenden. Hierfür wird die Klasse *cXPOpenCombo* verwendet.

Ein Objekt der Klasse *cXPOpenCombo* kann in der Klasse *cAppNavBar* in der Klassenbibliothek *Appl.vcx* gespeichert werden. Wenn dieses Objekt in der Standardsymbolleiste existiert, wird eine Referenz auf dieses Objekt in der Eigenschaft *goProgram.oXPOpenCombo* gespeichert.

Eine Referenz auf diese Combobox, die in der Standardsymbolleiste instanziiert wird, wird in der Eigenschaft *goProgram.oXPOpenCombo* des Anwendungsobjekts gespeichert.

Diese Combobox wird nur für Benutzer angezeigt, deren Benutzerstufe den Wert überschreitet, der in der Eigenschaft *nUserLevel* angegeben ist. Wenn der Wert der Eigenschaft *nUserLevel* = 0 ist, wird die Combobox für alle Benutzer angezeigt. Für Benutzer deren Benutzerstufe gleich oder kleiner als der in dieser Eigenschaft angegebene Wert ist, wird der normale Öffnen Dialog angezeigt. Die Benutzerberechtigungen werden berücksichtigt.

Die in der Combobox anzuzeigenden Formulare werden aus der Tabelle *Vfxfopen* gelesen. Die Anzeigereihenfolge wird über das neue Tabellenfeld *TbrCboSort* gesteuert. Wenn der Wert dieses Feldes 0 ist, wird das Formular nicht in der Combobox angezeigt.

Die Combobox enthält zwei Spalten, von denen nur die erste Spalte sichtbar ist und die Namen der auswählbaren Formulare enthält. Die Formularnamen werden aus dem Feld *Title* aus der Tabelle *Vfxfopen* gelesen. Die Anzeigereihenfolge in der Combobox wird durch die Werte des Feldes *TbrCboSort* festgelegt. Die zweite, nicht sichtbare, Spalte der Combobox enthält den auszuführenden Befehl.

Der auszuführende Befehl wird von der Methode *ItemExecute()* verarbeitet.

Eigenschaft

nUserLevel – Die Combobox wird in der Standardsymbolleiste nur für Benutzer angezeigt, deren Benutzerstufe größer als der Wert von dieser Eigenschaft ist. Wenn der Wert dieser Eigenschaft 0 ist, wird die Combobox für alle Benutzer angezeigt.

Methode

EnableCombo() – Anzeige der Combobox und füllen mit Werten.

DisableCombo() – Verbergen der Combobox.

ItemExecute() – Ausführen der Aktion, die mit dem Eintrag verbunden ist. Hier wird ein Formular gestartet oder es wird ein Befehl ausgeführt.

Filterdialog

Als neuer Operator steht jetzt „Beginnt mit“ zur Verfügung. Dieser Operator filtert nach Datensätzen, deren Wert im gewählten Feld mit dem eingegebenen Wert beginnt. Der Operator „Beginnt mit“ steht nur bei Feldern vom Typ Zeichen oder Memo zur Verfügung. Das Verhalten dieses Operators entspricht dem bisherigen Operator „Gleich“ für Felder vom Typ Zeichen oder Memo.

Das Verhalten des Operators „Gleich“ ist so geändert, dass bei Vergleichen von Zeichenketten ein genauer Vergleich durchgeführt wird.

Hilfe

Zu jedem Formular kann eine eigene kontextsensitive Hilfedatei angezeigt werden. Wenn ein Hilfethema nicht gefunden werden kann, wird automatisch die Startseite der Hilfedatei angezeigt.

Der Name der Hilfedatei wird in der Eigenschaft *cFormHelpFile* von Formularen angegeben. Die Hilfedatei wird aus dem gleichen Ordner geöffnet, in dem sich auch die Standardhilfedatei befindet. Der Pfad und Dateiname der Standardhilfedatei ist in der Eigenschaft *goProgram.cHelpFile* angegeben.

IFixField für Comboboxen

Eine Combobox kann so eingestellt werden, dass eine Auswahl oder Eingabe nur bei der Neuanlage eines Datensatzes möglich ist. Nach dem ersten Speichern eines Datensatzes bleibt die Combobox disabled. Dafür ist der Wert der Eigenschaft *IFixField* auf .T. einzustellen. Der Standardwert ist .F.

Auswahllisten

Für Auswahllisten kann die Breite der Spalten und die Sortierfolge voreingestellt werden. Diese Voreinstellungen des Entwicklers überschreiben in jedem Fall die Benutzereinstellungen.

Als Datenquelle für den Auswahllistendialog werden jetzt parametrisierte Cursoradapter unterstützt. Hierfür müssen einige Eigenschaften des Cursoradapters eingestellt werden.

cIdFieldFullName – Name des Feldes mit dem Wert des Parameters. Der Wert wird im Init Ereignis des Auswahllistendialogs in der Datensitzung des aufrufenden Formulars ausgewertet und in der Eigenschaft *vIDValue* zur späteren Verwendung in der Datensitzung des Auswahllistenformulars gespeichert.

cPickWhereClause – Where Klausel zur Verwendung im Auswahllistendialog. In der Regel wird in der Where Klausel der Wert *This.vIDValue* verwendet.

Start von Child-Formularen aus Childgrids

Child Formulare können jetzt aus Childgrids heraus gestartet werden. Das Child Formular wird beim Bewegen des Satzzeigers im Childgrid des Parent Formulars synchronisiert.

VFX – Parent/Child Builder

Im VFX – Parent/Child Builder gibt es eine neue Combobox „Initial Selected Alias“. Diese Combobox ist nur dann sichtbar, wenn das Formular auf der Klasse *cOneToMany* basiert oder Childgrids enthält.

Die Combobox enthält die Aliasnamen aus den *Recordsource* Eigenschaften der Childgrids aus dem Formular. Der in der Combobox ausgewählte Aliasname wird als Parentalias verwendet, wenn das Child Formular gestartet wird.

Ein Child Formular kann über den Onmore Dialog oder durch einen Doppelklick auf ein Steuerelement in einem Childgrid gestartet werden. Der im Dbclick Ereignis von Steuerelementen im Childgrid erforderliche Code wird vom VFX – Cchildgrid Builder eingefügt.

Es ist möglich aus einem Childgrid mehr als ein Child Formular zu starten. In diesem Fall wird der Onmore Dialog angezeigt, so dass der Benutzer ein Child Formular auswählen kann. In dem Onmore Dialog werden nur Child Formulare angezeigt, die zum aktuellen Childgrid gehören. Wenn zum aktuellen Childgrid nur ein Child Formular gehört, wird das Child Formular direkt angezeigt, ohne dass der Onmore Dialog angezeigt wird.

Wenn der Parent Teil eines Onetomany Formulars den Fokus hat, werden bei Aufruf des Onmore Dialogs alle Child Formular angezeigt, die zum Parent Teil gehören.

Ein Beispiel befindet sich in der Beispielanwendung VFX100Test im Formular OneToManyPageFrame.

Erweiterter VFX –Parent/Child Builder

Die Ausdrücke für Record Position Expression und Child Form Caption Expression werden jetzt im Parent Formular evaluiert. Dadurch ist es möglich in diesen Ausdrücken Feldnamen und Eigenschaften des Parent Formulars zu verwenden.

Die Ausdrücke müssen in einem Format vorliegen, das von der Funktion EVALUATE() interpretiert werden kann. Zeichenketten müssen in Anführungszeichen gesetzt werden,

Der Text und die Beschreibung für den Öffnen-Dialog werden ebenfalls im Parent Formular evaluiert. Hierdurch ist eine einfachere Lokalisierung dieser Texte möglich.

Hier ein Beispiel:

VFX - Extended Parent/Child Builder

Parent Form: Auto Sync. Child Form Close Child Form on Exit

The caption will be evaluated. Include constant text in quotation marks.

Command Type	Child Form	...	Parent field	Child field
Child Form	CHILD.SCX		Parent.PARENTID	Child.PARENTID

Onmore | Advanced | Help

Child Form:

Parent Field (Fix Field Value):

Child Field (Fix Field Name):

Child Form Caption Expression:

Text for Open Form Expression:

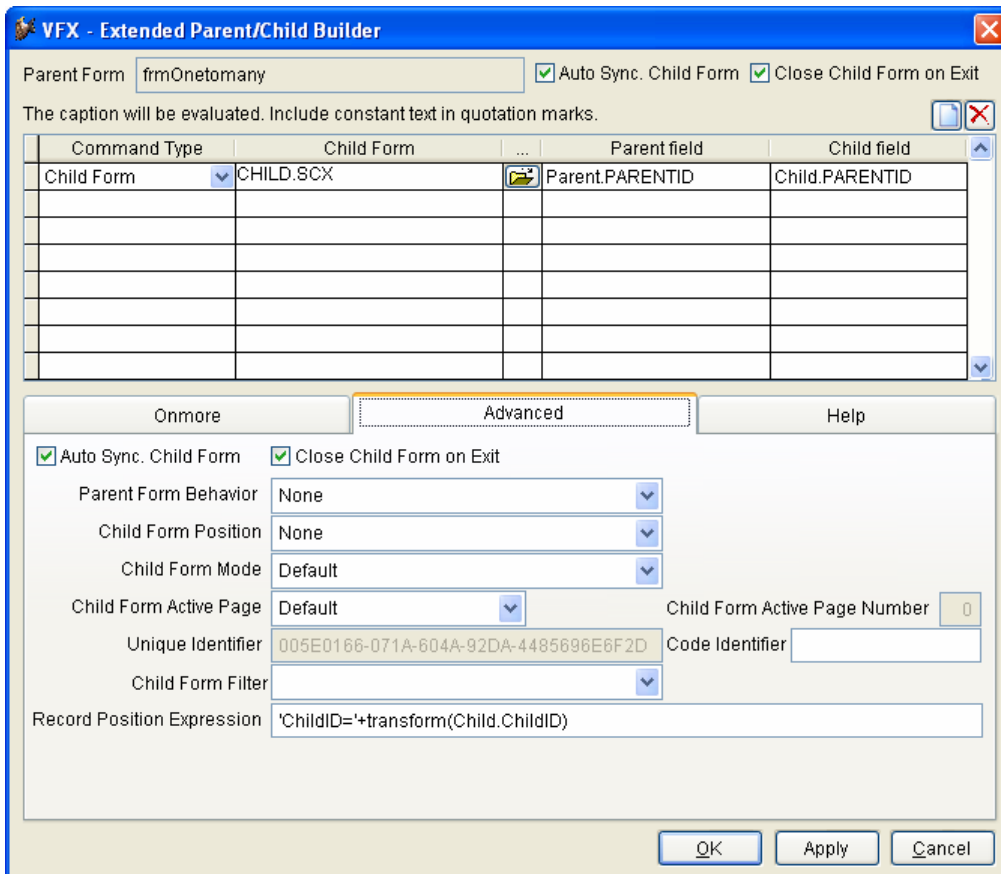
Description for Open Form Expression:

Available on onMoreDialog

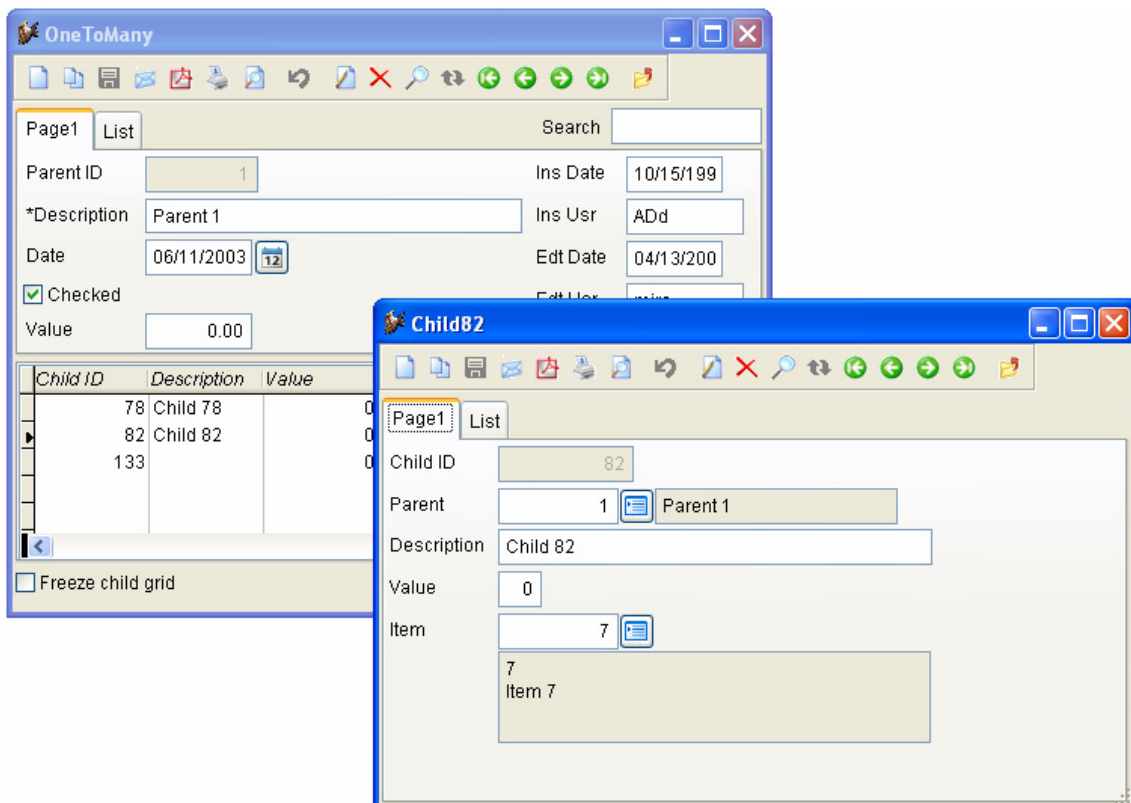
Initial Selected Alias:

OK Apply Cancel

In diesem Beispiel enthalten die Ausdrücke für Text for Open Form Expression und Description for Open Form Expression Code um lokalisierte Zeichenketten zu erhalten.



Dieses Beispiel befindet sich in VFX100Test. Das Parent Formular ist OneTo.scx. Das Child Formular wird für den im Childgrid aktuellen Datensatz geöffnet.



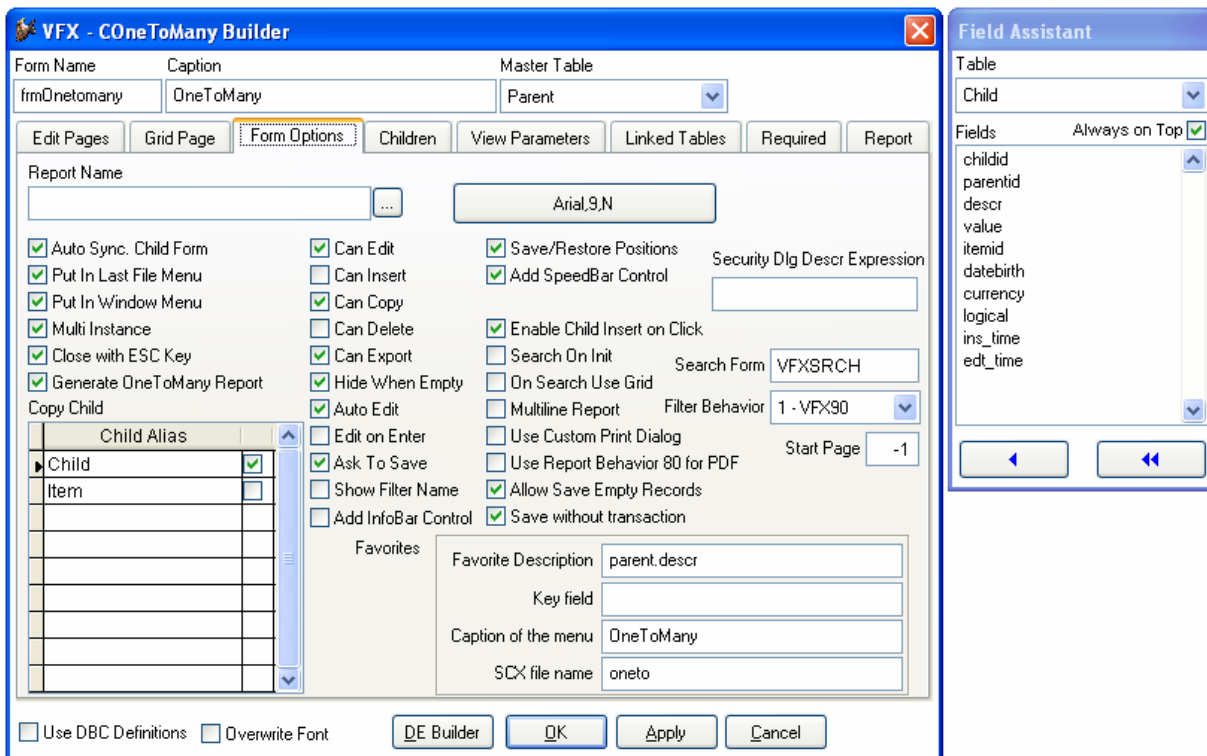
Neue Eigenschaften in der Formulkasse cOneToMany

1:n Berichte

In Formularen basierend auf der Klasse *cOnetomany* gibt es die neue Eigenschaft *lGenerateOneToManyReport*. Diese Eigenschaft steuert die Generierung von 1:n Berichten. Der Standardwert dieser Eigenschaft ist *.F.* um die Kompatibilität mit bestehenden Anwendungen zu erhalten.

Außerdem wird dieses Verhalten durch die Eigenschaft *nGenerateOneToManyReport* des Anwendungsobjekts gesteuert. Mit dieser Eigenschaft wird die Verwendung von 1:n Berichten in allen Onetomany Formularen der Anwendung beeinflusst. (0 – Formulareinstellung verwenden, 1 – In allen Formularen werden 1:n Berichte verwendet, 2 – 1:n Berichte werden in keinem Formular erstellt.) Der Wert der Eigenschaft *nGenerateOneToManyReport* kann im VFX – Application Builder eingestellt werden.

Der Wert der Eigenschaft *lGenerateOneToManyReport* kann in den Buildern VFX – COneToMany Builder, VFX – COneToManyPageFrame Builder und VFX – CTreeOneToMany Builder auf der Seite *Options* mit dem Kontrollkästchen *Generate OneToMany Report* eingestellt werden.

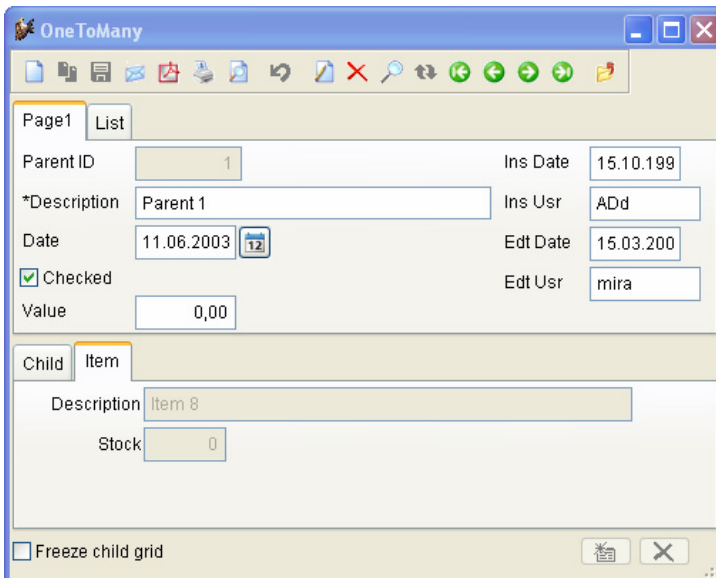


Bearbeitungsseiten für Child Daten

Zur Laufzeit werden jeder Seite eines Seitenrahmens *pgfChildGrid* auf Onetomany Formularen sowie *pgfPageFrame* auf OneToManyPageFrame Formularen die Eigenschaften *lCanUpdate* und *lEditing* hinzugefügt.

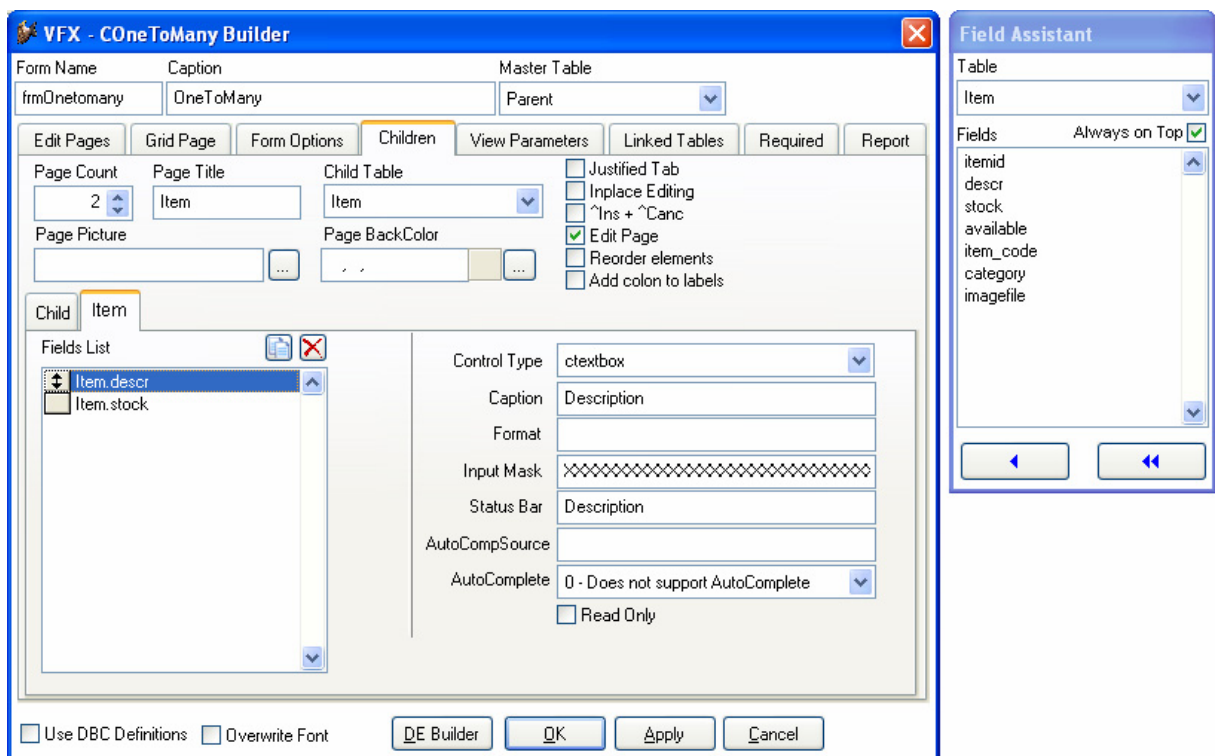
lEditing – Mit dieser Eigenschaft kann eingestellt werden, ob die Steuerelemente auf einer Bearbeitungsseite bearbeitet werden können. Wenn der Wert dieser Eigenschaft auf *.F.* eingestellt ist, sind alle Steuerelemente auf der Seite disabled.

lCanUpdate – Mit dieser Eigenschaft kann eingestellt werden, ob Child Datensätze hinzufügt und gelöscht werden können. Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, sind die Schaltflächen zum Einfügen und Löschen disabled.



Die Werte dieser Eigenschaften werden zur Laufzeit aus der Eigenschaft *cPagesSettings* des Seitenrahmens gelesen.

Die Einstellungen für diese Eigenschaften können im VFX - COneToMany Builder, VFX - COneToManyPageFrame Builder und VFX - CTreeOneToMany Builder auf den Seiten zur Bearbeitung von Child Daten mit den Kontrollkästchen „Inplace Editing“ und „^Ins + ^Canc“ gemacht werden.



Format der Eigenschaft *cPagesSettings*

Für jede Seite eines Seitenrahmens enthält der Wert der Eigenschaft *cPagesSettings* eine Zeile. Diese Zeile hat das folgende Format:

```
<nPageType>_ <cAlias>;<lEditing>;<lCanUpdate>
```

nPageType – Gibt den Typ der Seite an: 0 – Parent Seite, 1 – Bearbeitungsseite für Child Daten, 2 – Grid-Seite für Child Daten.

cAlias – Enthält den Alasnamen für eine Child Seite. Bei Parent Seiten ist dieser Wert leer.

lEditing – Enthält den Wert der Eigenschaft *lEditing* von Bearbeitungsseiten für Child Daten. Bei Grid Seiten für Child Daten ist dieser Wert leer.

lCanUpdate - Enthält den Wert der Eigenschaft *lCanUpdate* von Bearbeitungsseiten für Child Daten. Bei Grid Seiten für Child Daten und Parent Bearbeitungsseiten ist dieser Wert leer.

Beispiel:

```
2_ child;;
1_ item;.F.;.F.
```

Die Klasse cComboPicklist

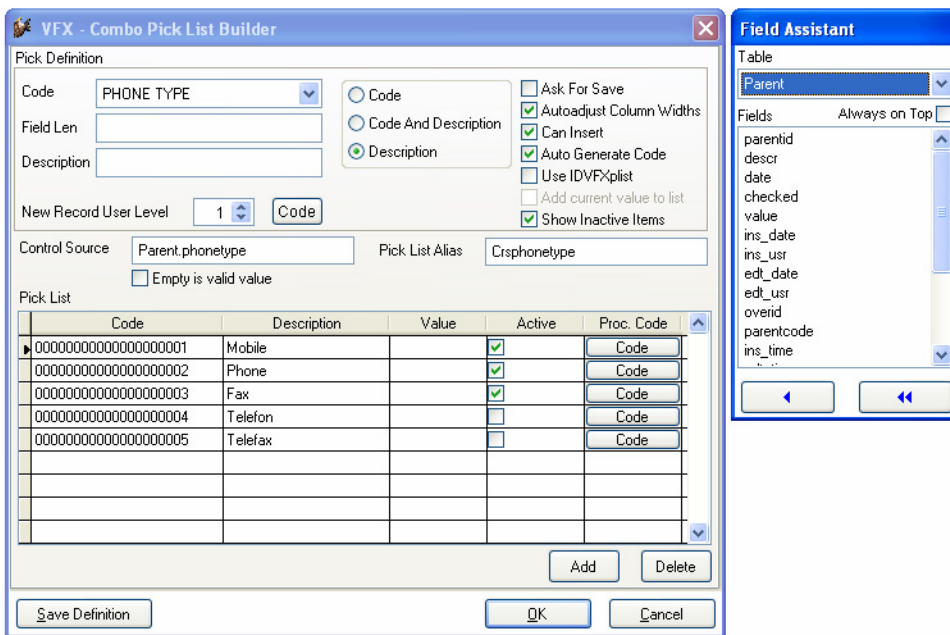
Für Combobox Auswahllisten, die kein Pflichtfeld sind, steht im Kontextmenü die neue Option „zurücksetzen“ zur Verfügung. Wenn diese Option zur Laufzeit ausgewählt wird, wird der Inhalt der Combobox gelöscht.

Wenn die Combobox so eingestellt ist, dass nur der Code angezeigt wird, kann mit der Eigenschaft *lAddCurrentValueToList* eingestellt werden, dass der eingegebene Wert automatisch der Auswahlliste hinzugefügt wird, wenn der Wert noch nicht in der Auswahlliste enthalten ist. Dieses Verhalten kann im VFX – Ccombopicklist Builder mit dem Kontrollkästchen *Add current value to list* eingestellt werden.

Die Verarbeitung wird in der Methode *FillItems* durchgeführt, die von der Methode *Requery* der *cComboPicklist* Klasse aufgerufen wird.

Wenn die Ccombopicklist so eingestellt ist, dass zur Laufzeit neue Einträge der Auswahlliste hinzugefügt werden können, wird bei der Initialisierung eine Referenz auf die Ccombopicklist dem Array *aAdditionalControlsToRequery* des Formulars hinzugefügt.

Mit der Eigenschaft *lShowInactiveItems* kann eingestellt werden, ob nicht aktive Werte angezeigt werden sollen. Wenn der Wert dieser Eigenschaft auf *.F.* eingestellt ist, werden nicht aktive Einträge nicht in der Combobox angezeigt. Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, werden nicht aktive Einträge in der Combobox disabled angezeigt. Im VFX – Ccombopicklist Builder kann diese Eigenschaft mit dem Kontrollkästchen *Show Inactive Items* eingestellt werden.



Eigenschaften

lAddCurrentValueToList – Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist und der eingegebene Wert noch nicht in der Auswahlliste enthalten ist, wird der Wert automatisch der Auswahlliste hinzugefügt. Der Wert dieser Eigenschaft wird nur berücksichtigt, wenn nur der Code angezeigt wird.

lShowInactiveItems – Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, werden in der Auswahlliste auch nicht aktive Werte angezeigt.

nColumnCnt – Diese Eigenschaft wird intern verwendet. Der Wert dieser Eigenschaft entspricht dem Wert des Feldes *ColumnCnt* aus der Tabelle *VfxPDef*.

IIDVfxPlist – Diese Eigenschaft wird intern verwendet. Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist und die Tabelle *VfxPlist* das Feld *IdvfxPlist* vom Typ Integer Autoinc enthält, wird dieses Feld für die Eigenschaft *BoundColumn* der Combobox verwendet.

Methoden

RightClick – Erstellt ein Kontextmenü mit den Einträgen Ausschneiden, Kopieren, Einfügen und Zurücksetzen. Die Option zurücksetzen steht zur Verfügung, wenn die Controlsource der Combobox kein Pflichtfeld ist.

ResetValue – Ersetzt den Wert der ControlSource mit einem leeren Wert. Diese Methode wird vom Eintrag zurücksetzen aus dem Kontextmenü aufgerufen.

Die Klasse cFooterBar

Die Klasse *cFooterBar* befindet sich in der Klassenbibliothek *VfxCtrl.vcx*. Diese Klasse ist so ähnlich wie die Klasse *cInfoBar* aufgebaut. Die Klasse *cFooterBar* zeigt Anwendern am unteren Rand von Formularen Informationen über den aktuellen Datensatz an. Innerhalb dieser Container-Klasse können Steuerelemente platziert werden, die Informationen anzeigen. Es ist möglich auf einem Formular mehr als einen *cFooterBar* Container zu platzieren.

Wenn die Größe des Formulars geändert wird, bleiben *cFooterBar* Container stets am unteren Formularrand und nur die Breite wird an die Breite des Formulars angepasst. Für die Änderung der Größe wurde der Klasse *cDataFormVfxbase* eine Eigenschaft hinzugefügt:

oFooterControl – Diese Eigenschaft enthält eine Referenz auf den obersten *cFooterBar* Container auf einem Formular.

Bei den Steuerelementen in den *cFooterBar* Containern sollte der Wert der Eigenschaft *Comment* = “<NORESIZE>” eingestellt werden.

Die *cFooterBar* Container werden dem Array *aAdditionalControls* des Formulars hinzugefügt und beim *Refresh* Ereignis des Formulars automatisch mit aktualisiert.

Die Klasse cMapPoint

Diese Klasse kapselt die Ansteuerung von Microsoft Map Point.

Eigenschaften

nUnits – Ganzzahl für die Einheit der Entfernung. 0 – Meilen, 1 – Kilometer.

oMapPoint – Nur intern verwendet. Diese Eigenschaft enthält eine Referenz auf das MapPoint Objekt.

oMap – Nur intern verwendet. Diese Eigenschaft enthält eine Referenz auf das MapPoint Objekt und wird zum Auffinden von Adressen verwendet.

oRoute – Nur intern verwendet. Diese Eigenschaft enthält eine Referenz auf ein Streckenobjekt zur Berechnung von Entfernungen zwischen Adressen.

Methoden

OpenMapPoint(tlForceNewInstance, tnUnits) – Diese Methode startet MapPoint und speichert eine Referenz auf ein Map Objekt in der Eigenschaft *oMap*.

tlForceNewInstance – Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, wird auf jeden Fall eine neue Instanz von MapPoint gestartet. Wenn der Wert dieser Eigenschaft auf *.F.* eingestellt ist, wird eine vorhandene Instanz verwendet, wenn MapPoint bereits läuft.

tnUnits – Einheit für Entfernungen. 0 – Meilen, 1 – Kilometer.

FindAddress(tcStreet, tcCity, tcOtherCity, tcRegion, tcPostalCode, tvCountry) – Suche nach der Adresse entsprechend den angegebenen Parametern. Diese Methode liefert eine Sammlung von Orten, die möglicherweise zu den Parametern passen. Dieser Methode muss mindestens ein Parameter übergeben werden. Wenn ein Parameter nicht für eine Adresse unterstützt wird, zum Beispiel eine Region für eine Adresse in Deutschland, wird dieser Parameter ignoriert. Der Rückgabewert ist eine Sammlung *FindResults*.

tcStreet – Der Straßename oder die zu findende Straßenadresse.

tcCity – Der Name der Stadt.

tcOtherCity – Der Name der zweiten zu findenden Stadt. Diese Angabe wird nur für Adressen in Großbritannien benötigt.

tcRegion – Der Name der Region.

tcPostalCode – Die Postleitzahl der zu findenden Adresse.

tvCountry – Das zu findende Land. Eine Liste mit gültigen Werten für diese Eigenschaft ist auf dieser Website zu finden:

<http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/mappoint2004/BIZOMVGeoCountry.asp>

Anwendungsbeispiel:

```
loMapPoint = NEWOBJECT("cMapPoint")
loMapPoint.OpenMapPoint(.T., 1)
loResults = loMapPoint.FindAddress(lcStreet, lcTown, , , lcZIP,
GeoCountryCode)
lnGeoQuality = loResults.ResultsQuality
loLocation = loResults.Item(1)
lnGeoLong = loLocation.Longitude
lnGeoLat = loLocation.Latitude
```

GetLocation(tnLatitude, tnLongitude) – Diese Methode liefert ein Ortsobjekt zu den als Parameter übergebenen Werten für Breitengrad und Längengrad. Das Ortsobjekt kann für einen Ort, eine Adresse oder geografische Koordinaten sein.

RouteAddWayPoint(tnLatitude, tnLongitude, toWayPoint) – Diese Methode fügt einen Wegpunkt der aktuellen Strecke (bzw. dem aktuellen WayPoint Objekt) hinzu.

Breitengrad (tnLatitude) und Längengrad (tnLongitude) als Parameter übergeben.

Ortsobjekt (toWayPoint) als Parameter übergeben.

Das Ortsobjekt kann erstellt werden mit:

Aufruf der Methode *GetLocation*.

Aufruf der Methode *Findaddress*.

Anwendungsbeispiel:

```
loMapPoint = NEWOBJECT("cMapPoint")
loMapPoint.OpenMapPoint(.T., 1)
loResults = loMapPoint.FindAddress(lcStreet, lcTown, , , lcZIP,
GeoCountryCode)
loLocation = loResults.Item(1)
loMapPoint.RouteAddWayPoint(loLocation)
```

RouteClear() – Diese Methode löscht Wegpunkte von der aktuellen Strecke.

RouteMoveWayPoint(tnItemNumber, toAnchor) – Verschieben eines Wegpunktes.

tnItemNumber – Nummer des Wegpunktes.

toAnchor – Ortsobjekt des neuen Ortes.

RouteCalculate() – Berechnung einer Wegstrecke anhand von zwei oder mehr Wegpunkten. Bevor diese Methode ausgeführt werden kann, müssen der Strecke mindestens zwei Wegpunkte hinzugefügt werden. Der Rückgabewert ist eine Referenz auf das Streckenobjekt. Nach Ausführung dieser Methode stehen die Werte in den Eigenschaften *Distance* und *DrivingTime* des Streckenobjekts zur Verfügung.

Anwendungsbeispiel:

```
loRoute = This.oMapPoint.RouteCalculate()
lnWayDistance = loRoute.Distance    && in distance units (km. or mile)
lnWayTime = loRoute.DrivingTime * 24    && in hours
```

CloseMapPoint() – Schließen von MapPoint.