



VFX 8.0 – Workshop

Juni 2004

© Dipl.-Inform. Uwe Habermann

Einführung

In diesem Workshop wird ein Überblick über die Leistungsfähigkeit des Frameworks Visual Extend gegeben. Den Teilnehmern werden an zahlreichen Beispielen die vielfältigen Möglichkeiten gezeigt. Dabei werden auch die von VFX 8.0 unterstützten Eigenschaften von VFP 8.0 gezeigt. Es wird gezeigt, wie schnell und einfach mit VFX 8.0 das Grundgerüst für eine Anwendung erstellt werden kann. Es werden Formulare mit Hilfe der VFX-Generatoren erstellt und bearbeitet. Wichtige VFX-Methoden und die damit möglichen Eingriffe in den Programmablauf werden vorgestellt.

In diesem Workshop erarbeiten wir uns die Möglichkeiten der neuen Version von Visual Extend zu verstehen und in eigenen Anwendungen zu nutzen.

Zu den neuen Funktionen gehören unter Anderem:

Unterstützung des Treeview-Controls in den VFX-Formularklassen. Schutz von VFX-Anwendungen durch Produktaktivierung, Teile der Anwendung können getrennt aktiviert werden. Nutzung aller Eigenschaften des Menüs von VFP 8 durch den neuen VFX-Menü-Designer. Erstellen von PDF-Dateien aus allen Berichtsausgaben. E-Mail-Versand von Berichtsausgaben. Mehr Eigenschaften bei Grid-Reports. Client-Database-Update jetzt auch beim Einsatz von Triggern problemlos. Client-Database-Update jetzt auch für SQL-Server! Verwendung der Klasse DataEnvironment von VFP 8. Kleine Makrosprache zum automatischen Download und zur Installation von beliebigen Programmen aus dem Internet. Und das ist längst nicht alles...

VFX – Was ist das?

Visual Extend 8.0 ist ein Builder-unterstütztes Framework. Es besteht aus einer Musteranwendung und aus reentranten Buildern. VFX ermöglicht es Anwendungen zu erstellen, die auf Tabellen als Datenquelle basieren oder eine Remote-Datenquelle verwenden. Die mit VFX erstellten Anwendungen sind in ihrer Bedienung weitgehend kompatibel zu den bekannten Office-Anwendungen. Es können Anwendungen in den Sprachen Deutsch, Englisch, Französisch, Italienisch und Spanisch erstellt werden.

Installation

VFX kann aus dem Internet herunter geladen werden. Es besteht aus einer ausführbaren Datei, die das mit InstallShield erstellte Setup enthält. VFX kann in einem beliebigen Ordner installiert werden. Nach der Installation befinden sich in diesem Ordner die leere VFX-Musteranwendung sowie ein Ordner „Builder“, in dem die Builder-Anwendungen von VFX gespeichert sind.

Um VFX sinnvoll einsetzen zu können, empfiehlt es sich das VFX-Menü gleich beim Start von VFP zu laden. Dazu muss die Anwendung Vfxmnu.app in einem Startprogramm oder direkt in der Konfigurationsdatei (config.fpw) ausgeführt werden.

Wenn VFP gestartet wird, wird das VFX-Menü in das VFP-Menü integriert und anschließend wird automatisch der VFX Application Manager geladen.

Beim ersten Start des VFX-Menüs tauscht VFX den Builder-Pfad von VFP aus. Die Systemvariable _Builder zeigt anschließend auf die VFX-Builder-Anwendung Vfxbldr.app. Außerdem wird der Suchpfad von VFP ergänzt, so dass zusätzlich im Builder-Ordner der VFX-Installation gesucht wird. In diesem Ordner befinden sich alle Builder von VFX.

Die Ausführung der VFX-Builder ist durch einen Aktivierungsschlüssel geschützt. Wenn VFX auf dem aktuellen Rechner noch nicht aktiviert wurde, erscheint bei jedem Aufruf einer VFX-Anwendung, dazu gehört auch das VFX-Menü, ein Aktivierungsfenster mit einer Registrierungsnummer. Einen zu der Registrierungsnummer passenden für 30 Tage gültigen Aktivierungsschlüssel kann man auf der Website www.visualextend.de kostenlos erhalten. Wer sich zum Kauf von VFX entschließt bekommt einen unbefristet gültigen Aktivierungsschlüssel.

Erstellen einer VFX 8.0-Anwendung

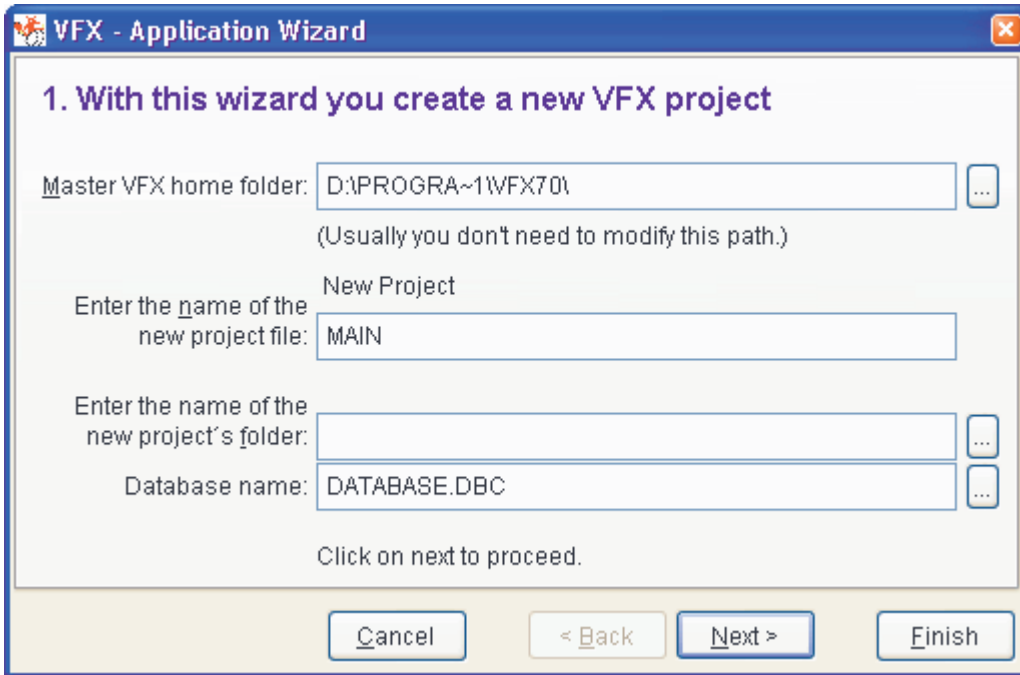
Ein neues VFX-Projekt kann innerhalb einer Minute angelegt werden. Wenn die Datenbank bereits vorhanden ist, können mithilfe der VFX-Formulargeneratoren in kurzer Zeit zahlreiche Standardbearbeitungsformulare erstellt werden. Ein Prototyp einer Anwendung ist so, ohne Programmierung, sehr schnell erstellt.

Die VFX-Anwendung kann jederzeit mit normalen VFP-Mitteln weiterentwickelt werden.

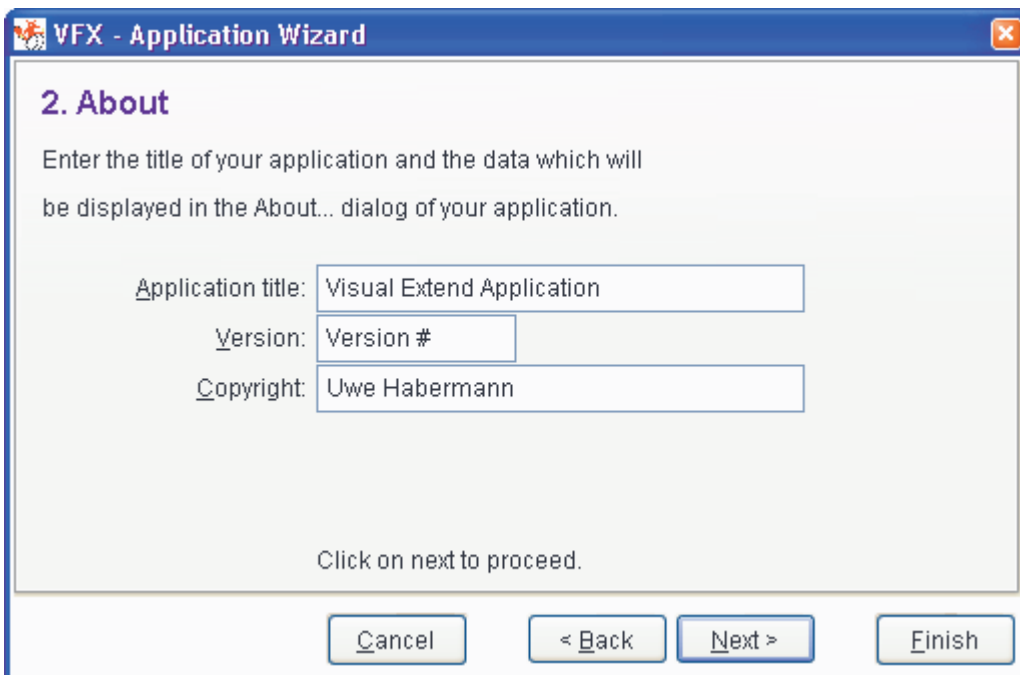
VFX – Application Wizard

Mit dem VFX – Application Wizard wird ein neues VFX Projekt angelegt. In einigen Dialogschritten wird der Entwickler aufgefordert die zum Erstellen des Projekts erforderlichen Angaben zu machen.

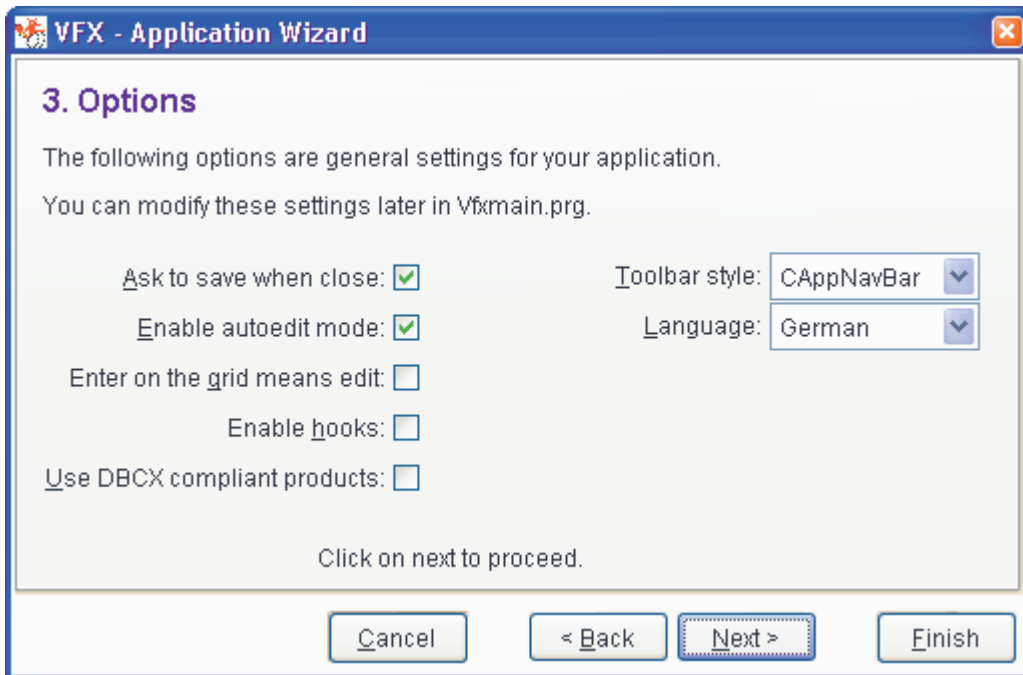
VFX bietet keine Werkzeuge zur Erstellung oder Bearbeitung der Datenbank. Der Entwickler kann also den VFP Datenbank-Designer oder ein Produkt eines Drittanbieters verwenden. Wenn bereits eine Datenbank vorhanden ist, sollte der Name dieser Datenbank im VFX – Application Wizard eingetragen werden.



Die auf der Seite About gemachten Angaben werden in der Include-Datei Usertxt.h gespeichert. Zur Laufzeit der Anwendung werden die Daten im About-Dialog angezeigt. Der Titel der Anwendung wird außerdem in der Caption der Anwendung angezeigt.



Im dritten Dialogschritt muss insbesondere die Sprache der zu erstellenden Anwendung angegeben werden. VFX kann Anwendungen in Deutsch, Englisch, Französisch, Italienisch und Spanisch erstellen.



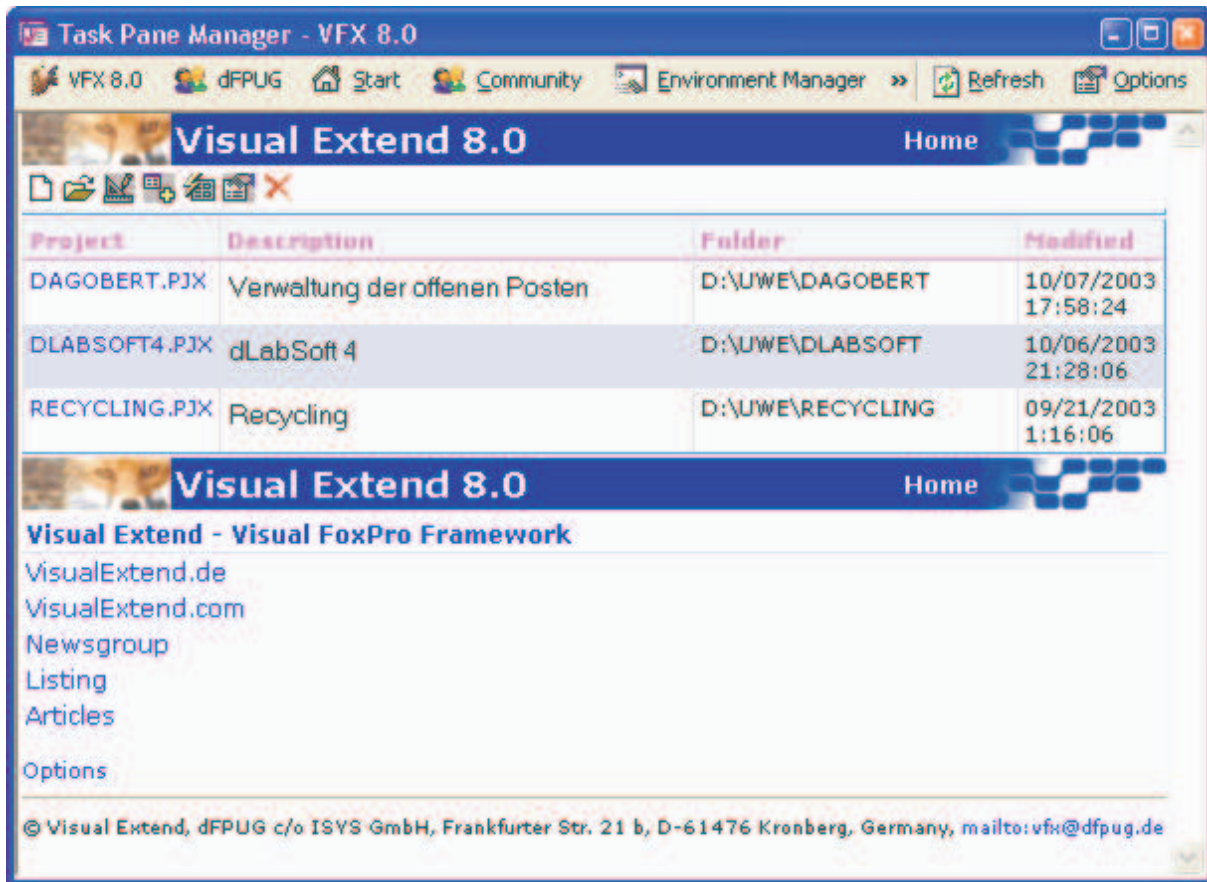
Die weiteren Optionen können bei Bedarf später in Vfxmain.prg geändert werden.

Auf der letzten Seite des Application Wizard werden schließlich Angaben gemacht, die in den Projektinformationen gespeichert werden.

Beim Erstellen des Projekts wird die VFX-Musterapplikation in den neuen Projektordner kopiert. Entsprechend der gewählten Sprache werden Include-Dateien und Menüs hinzugefügt. Zum Schluss wird das neue Projekt vollständig neu kompiliert und kann dann sofort ausgeführt werden.

VFX 8.0 Task Pane

Der VFX – Application Manager ist in die VFP Task Pane integriert.



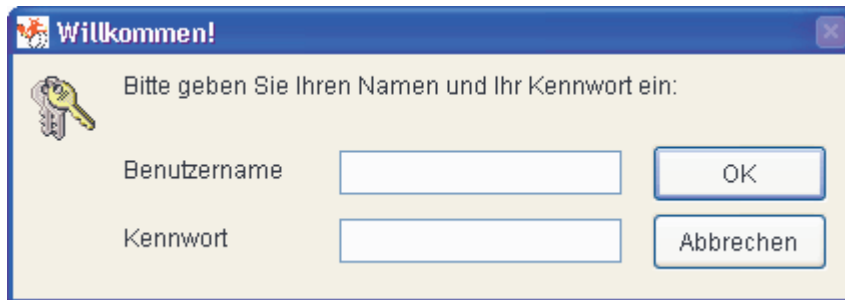
Über die Symbolleiste stehen folgende Funktionen zur Verfügung:

- New Project* Startet den VFX – Application Wizard.
- Open Project* Öffnet ein VFP-Projekt und stellt den aktuellen Pfad auf den Projektordner.
- Modify Project* Öffnet das in der VFX 8.0 Task Pane selektierte Projekt und stellt den aktuellen Pfad auf den Projektordner.
- Add Project* Fügt ein vorhandenes VFP-Projekt der VFX 8.0 Task Pane hinzu.
- Rebuild* Neu kompilieren aller Dateien des in der VFX 8.0 Task Pane selektierten Projekts. Das Projekt wird nach dem kompilieren zur Bearbeitung geöffnet.
- Properties* Start der VFX – Project Properties zum in der VFX 8.0 Task Pane selektierten Projekt.
- Delete* Entfernt das selektierte Projekt aus der VFX 8.0 Task Pane.

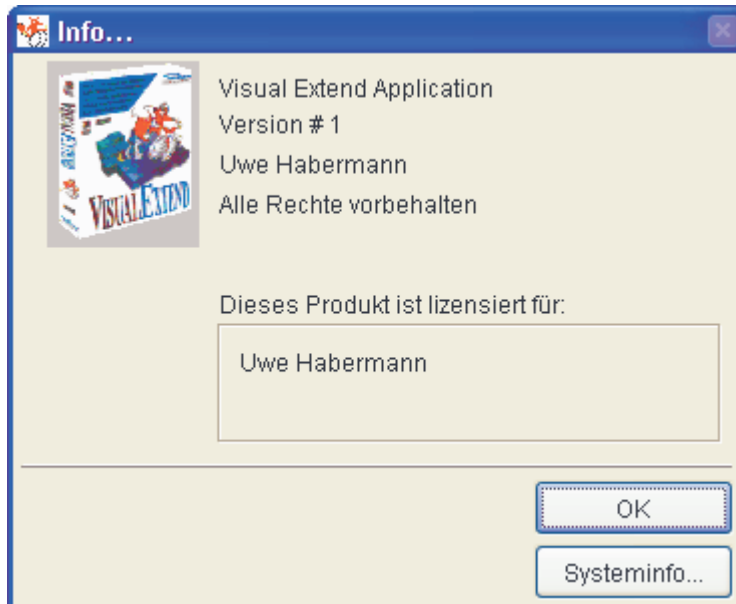
Der Vorteil des VFX – Application Manager ist, dass nicht nur das Projekt geöffnet wird, sondern dass auch das aktuelle Verzeichnis auf das Projektverzeichnis gesetzt wird. Der VFX – Application Manager sollte daher unbedingt zum Öffnen von Projekten verwendet werden.

Diskussion der erstellten VFX-Anwendung

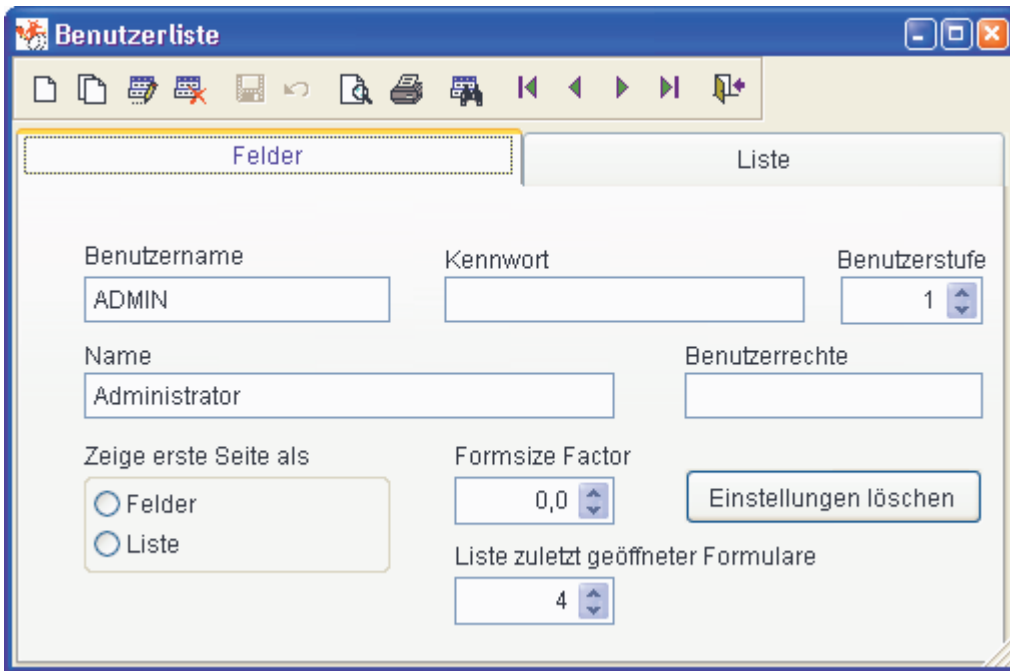
Die vom VFX – Application Wizard erstellte Anwendung hat schon viele gute Eigenschaften. Die Anwendung begrüßt den Benutzer mit einem Splash-Screen und einem Anmeldedialog.



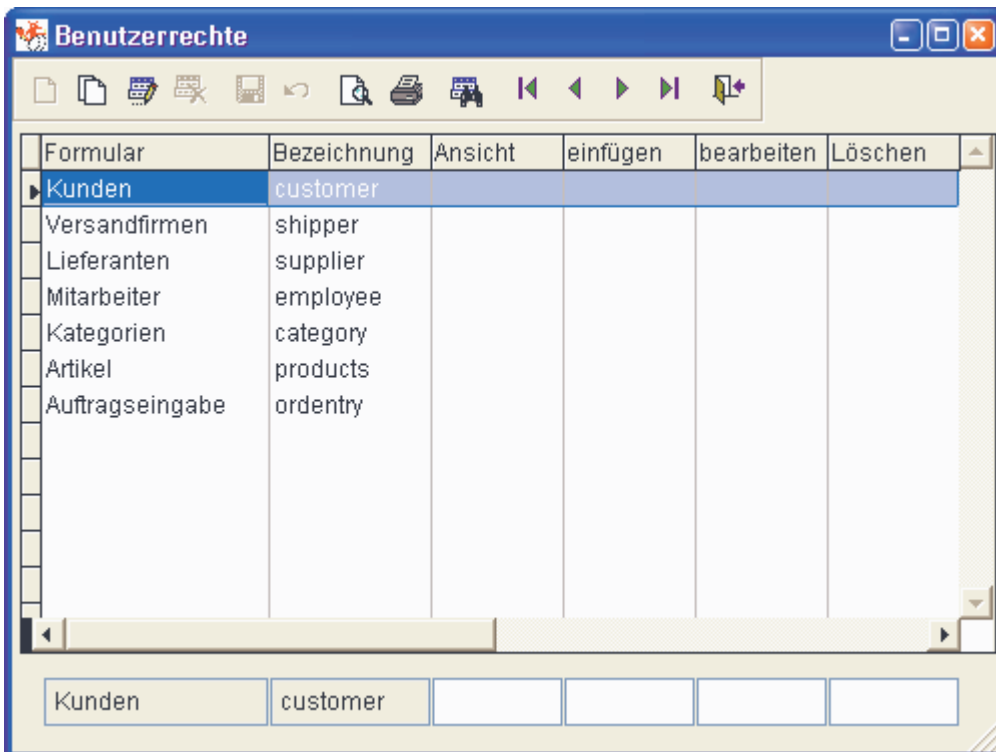
Die Anwendung hat einen About-Dialog, der über den Menüpunkt Hilfe, Info erreicht werden kann.



Eine Verwaltung der Benutzerdaten und der Benutzerrechte sind fertig integriert.



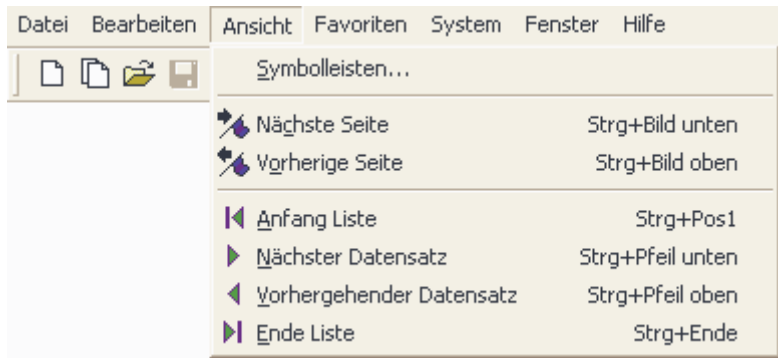
Die Zugriffsrechte werden über die Benutzerstufe gesteuert. Der Administrator hat die Benutzerstufe 1 und damit alle Rechte. Ein Benutzer, der die Benutzerstufe 99 hat, hat die wenigsten Rechte. Im Formular Benutzerrechte kann für jedes Formular festgelegt werden welche Benutzerstufe erforderlich ist um das Formular anzeigen zu können, um neue Datensätze erfassen zu können, um vorhandenen Datensätze bearbeiten zu können und um Datensätze löschen zu können.



Wenn ein Benutzer nicht das Recht hat ein Formular anzuzeigen, wird das betreffende Formular nicht instanziiert und erscheint nicht im Öffnen-Dialog.

Solange im Dialog Benutzerrechte keine Benutzerstufen eingetragen sind, gelten die Einstellungen, die mit dem VFX Form Wizard in den Formular-Eigenschaften *lcaninsert*, *lcancopy*, *lcanedit* und *lcandelete* hinterlegt sind.

Die Anwendung hat ein Menü und eine Symbolleiste, die an die bekannten Office-Anwendungen erinnern lassen. Die VFX-Menüs enthalten Icons.

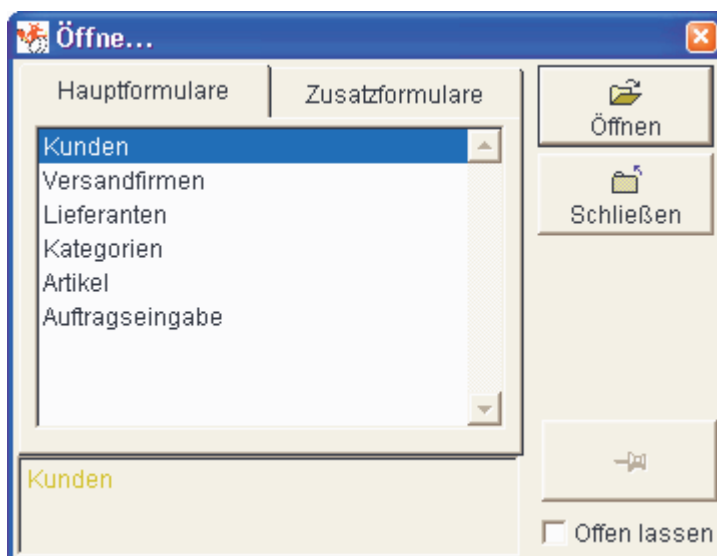


Die VFX 8.0-Symbolleisten erscheinen im „Hot Tracking“ Layout.



Öffnen-Dialog

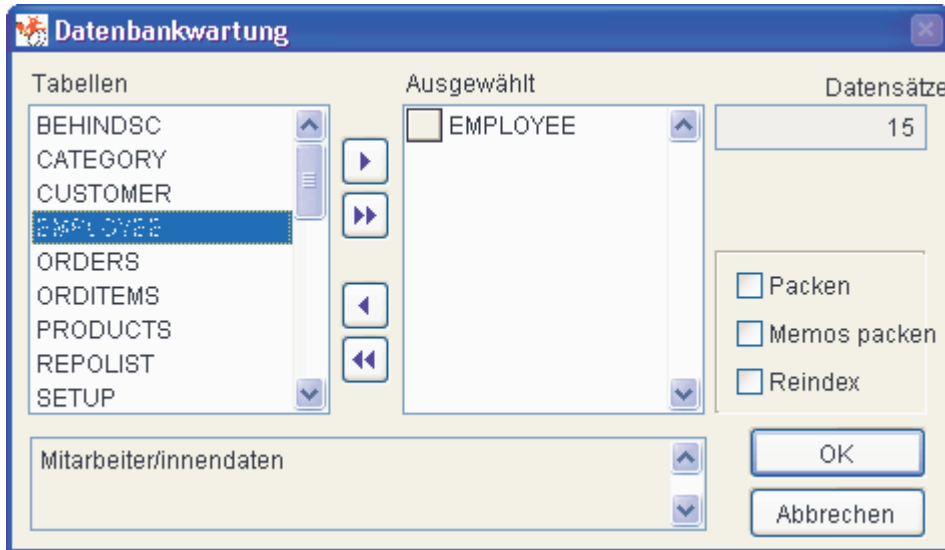
Der Öffnen-Dialog kann über das Öffnen-Symbol in der Symbolleiste oder über den entsprechenden Menüpunkt geöffnet werden und dient dem Start von Formularen.



Formulare, die mit den VFX – Form Builder erstellt wurden, werden automatisch im Öffnen-Dialog eingetragen. Der Öffnen-Dialog basiert auf den Daten der Tabelle Vfxopen.dbf.

Datenbankwartung

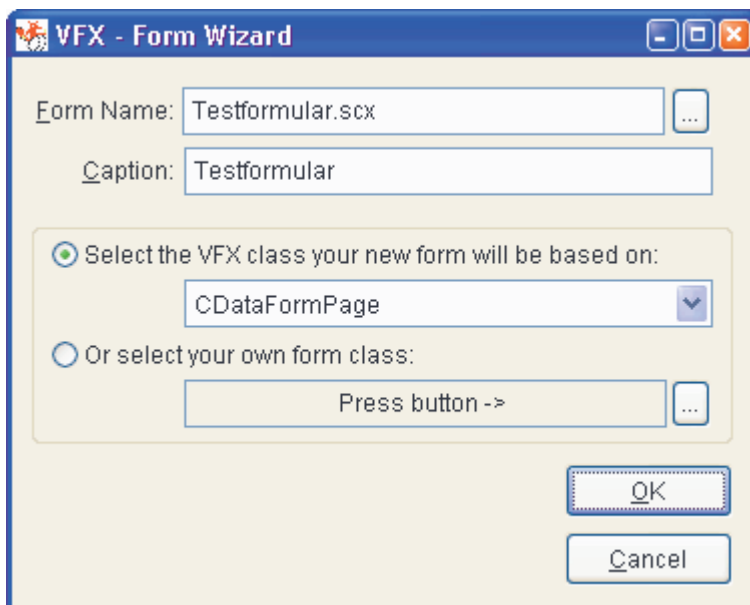
In einem „Mover“-Formular können Tabellen aus der verwendeten Datenbank ausgewählt werden. Die gewählten Tabellen können dann neu indiziert oder gepackt werden.



Erstellen von Formularen

VFX – Form Wizard

Mit dem VFX – Form Wizard werden neue Formulare erstellt.



Der Form Wizard fordert zur Eingabe eines Namens sowie des Titels für das zu erstellende Formular auf. Das Formular kann auf einer der VFX-Formularklassen oder, wahlweise, auf einer eigenen Formularklasse basieren. Die am häufigsten gebräuchlichsten VFX-Formularklassen sind CDataFormPage und CTableForm. Die erstellten Formulare werden im Form-Ordner unterhalb des aktuellen Projekts gespeichert.

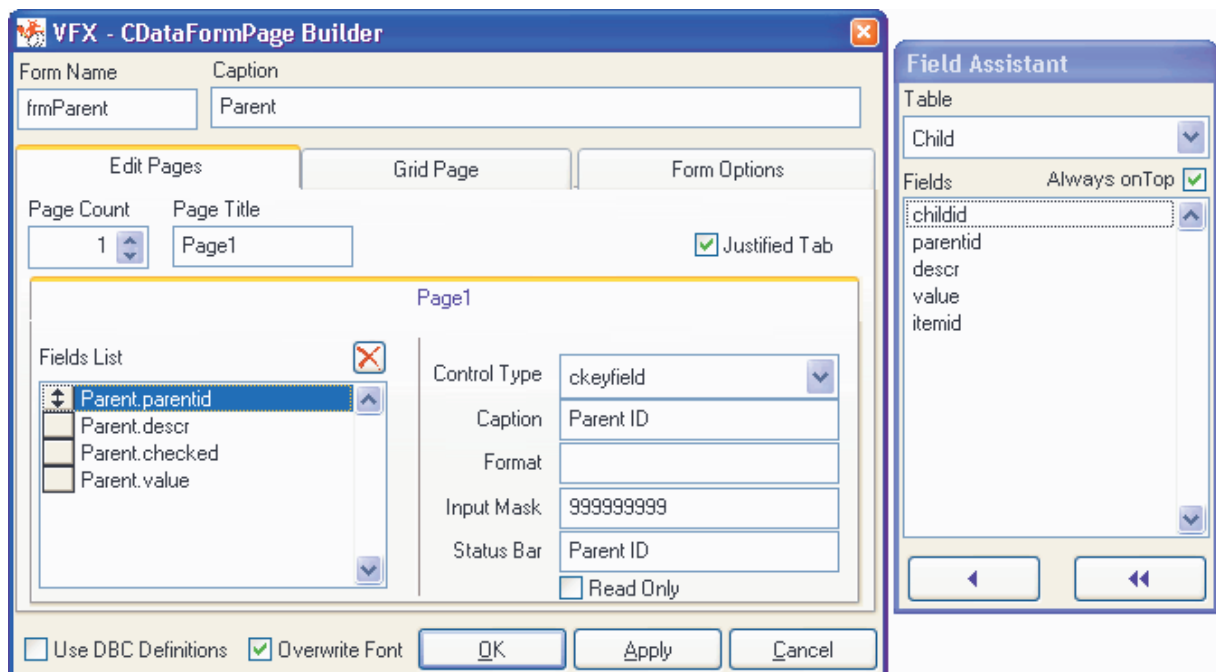
VFX - Form Builder

Die vom VFX – Form Wizard erstellten Formulare müssen nun mit Inhalten gefüllt werden. Dazu dienen die VFX – Form Builder. Es gibt Form Builder für Formulare basierend auf den VFX-Klassen:

- CDataFormPage – Standard-Datenbearbeitungsformular mit Seitenrahmen
- CTableForm – Datenbearbeitungsformular mit Grid und anderen Steuerelementen auf einer Seite
- COneToMany – Datenbearbeitungsformular mit Grid zur Bearbeitung von Child-Daten

Bevor jedoch mit einem VFX – Form Builder gearbeitet werden kann, muss die Datenumgebung des Formulars gefüllt werden. Mit normalen VFP-Mitteln sind die benötigten Tabellen und Ansichten der Datenumgebung hinzuzufügen.

VFX - CDataFormPage Builder

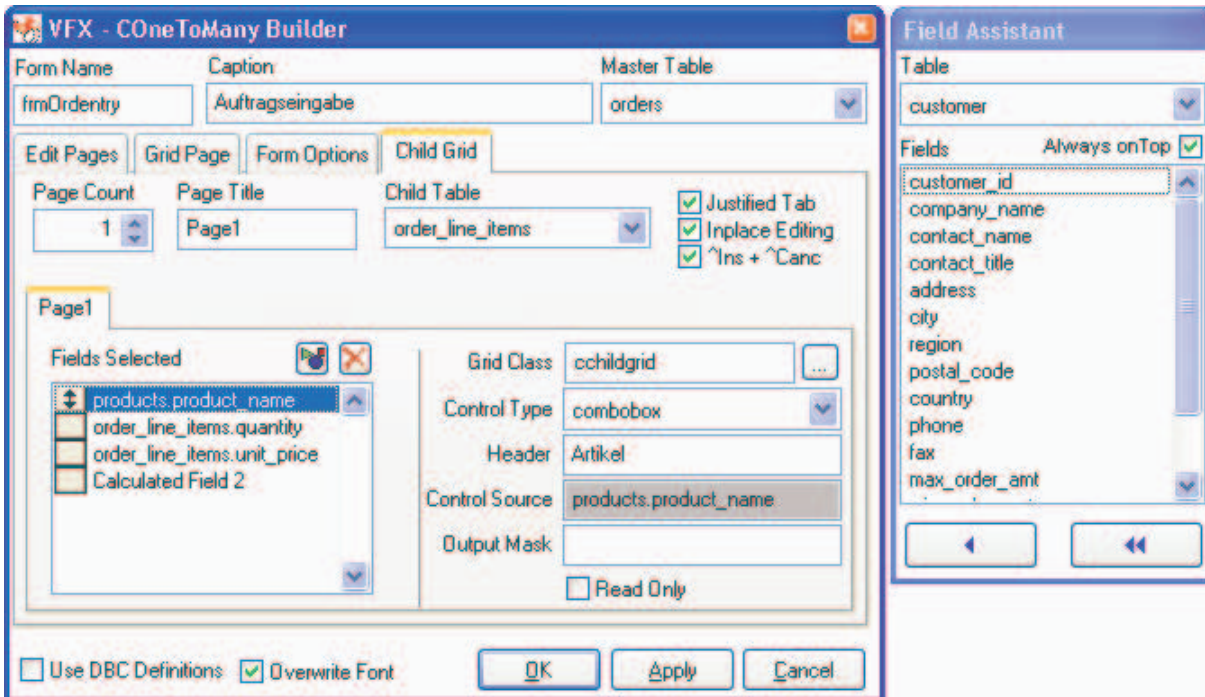


Jetzt kann der VFX – Form Builder gestartet werden. Alle VFX - Form Builder arbeiten reentrant, können also mehrmals auf einem Formular angewendet werden, ohne dass zuvor gemachte Einstellungen verloren gehen.

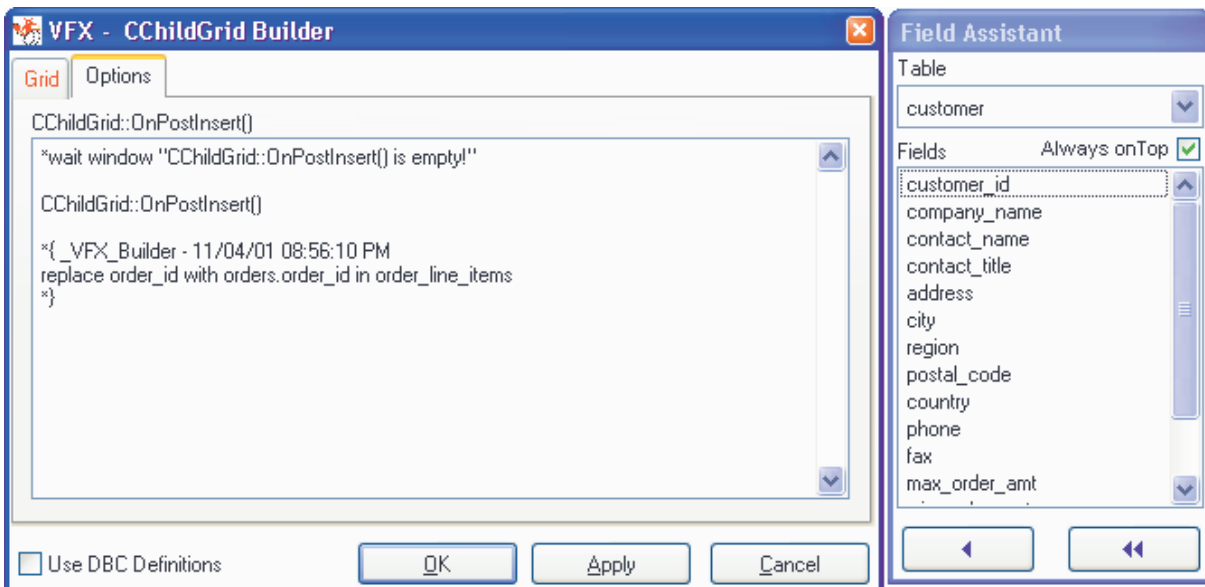
Im VFX - CDataFormPage Builder werden die Bearbeitungsseite und das Grid für die Suchseite eines Formulars erstellt. Auf der Seite „Options“ können Einstellungen an den Formular-Eigenschaften gemacht werden.

VFX - COneToMany Builder

Zusätzlich zu den Optionen, die der CDataFormPage Builder bietet hat der COneToMany Builder eine Seite zur Erstellung eines Childgrid.



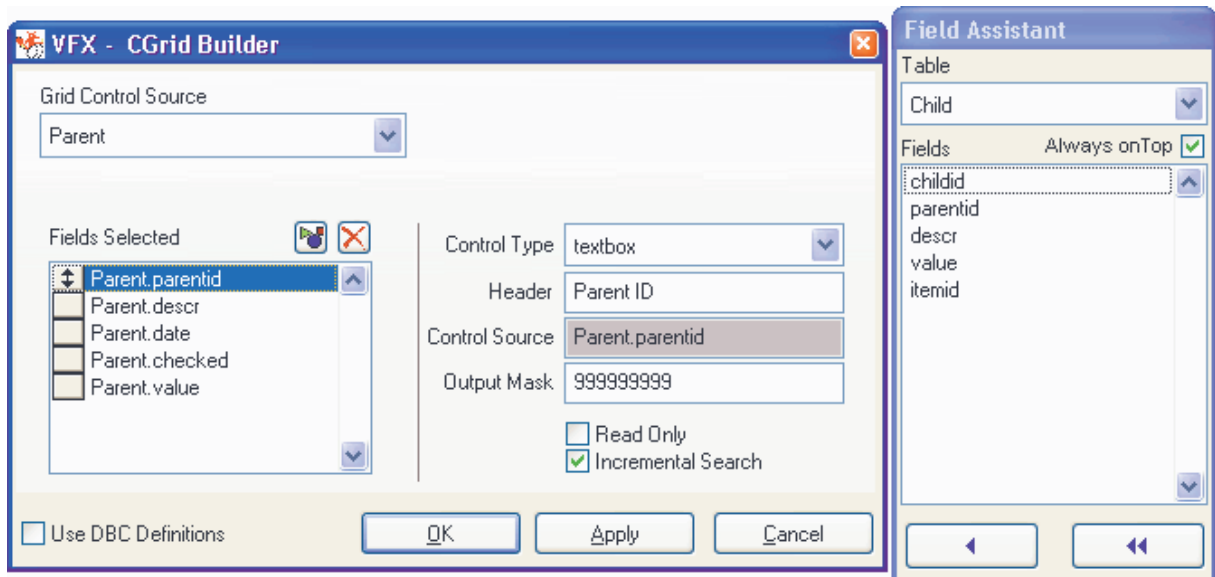
Bei OneToMany-Formularen ist zu beachten, dass zu jedem Child-Datensatz der Fremdschlüssel des Parent-Datensatzes hinzugefügt werden muss. Dies geschieht in der Methode onpostinsert des Childgrid. Der COneToMany Builder erstellt bereits einen Vorschlag für den Code, der den Fremdschlüssel einträgt. Der Code der Methode onpostinsert kann mit dem VFX CChildGrid Builder bearbeitet werden.



Der von VFX generierte Code ist mit einem Kommentarzeichen versehen. Zusätzlich ist eine Wait Window-Zeile enthalten, die den Entwickler darauf hinweist, dass der Code dieser Methode überprüft werden muss. Wenn der Fremdschlüssel ein einfacher Schlüssel ist, kann das Kommentarzeichen bedenkenlos entfernt werden, wie es auf dem Screenshot zu sehen ist. Wenn der Fremdschlüssel ein zusammengesetzter Schlüssel ist, muss der Code von Hand angepasst werden.

VFX - Grid Builder

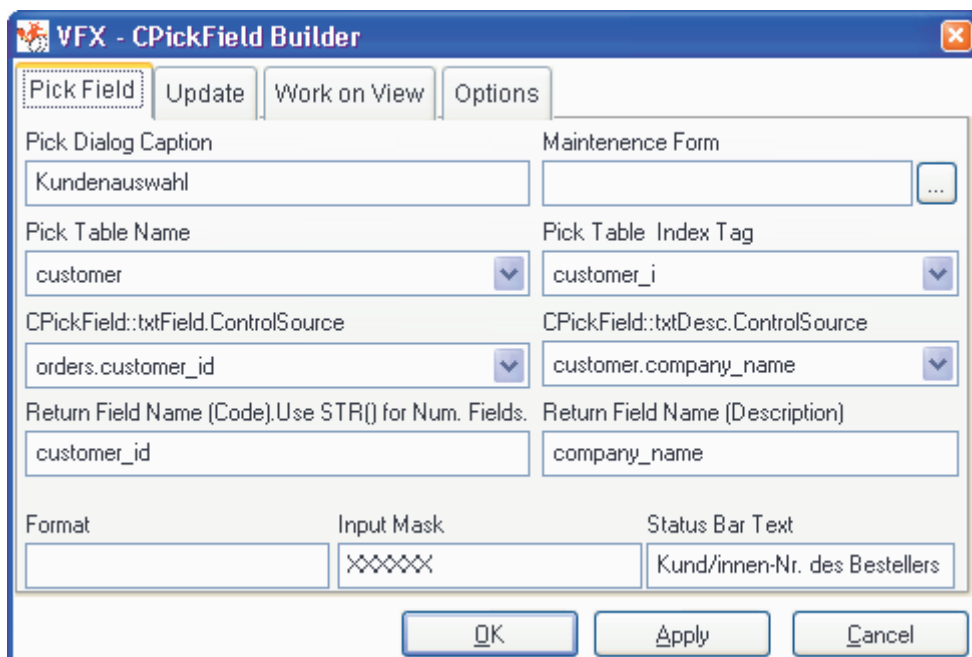
Die Funktionalität des VFX – Grid Builder ist im VFX – Form Builder enthalten. Damit jedoch nicht der VFX – Form Builder gestartet werden muss, wenn nur Änderungen im Grid gemacht werden sollen, steht der VFX – Grid Builder auch als eigenständige Anwendung zur Verfügung.



Auch der VFX – Grid Builder arbeitet reentrant, kann also mehrmals aufgerufen werden, ohne dass zuvor gemachte Einstellungen verloren gehen.

VFX – CPickField Builder

Eine leistungsfähige Eigenschaft von VFX ist die Verwendung von Auswahllisten. Ein Auswahlfeld, basierend auf der Klasse cPickField besteht aus einer Textbox, in die Werte eingegeben werden können, einer Schaltfläche zum Aufruf der Auswahlliste und aus einer weiteren Textbox in der Werte aus der Auswahltabelle angezeigt werden können. Alle für eine Auswahlliste benötigten Einstellungen können mit dem Builder gemacht werden.



Die „Pick Dialog Caption“ ist die Caption der Auswahlliste. Die „Maintenance Form“ ist der Name des Bearbeitungsformulars, in dem der Benutzer neue Daten der Auswahltabelle hinzufügen kann, wenn der gewünschte Wert noch nicht vorhanden ist.

„Pick Table Name“ ist der Name der Tabelle oder Ansicht auf der die Auswahlliste basiert. „Pick Table Index Tag“ ist der Indexschlüssel, der zur Validierung der Benutzereingabe verwendet wird.

In der nächsten Zeile des Builders werden die Controlsources der beiden im Auswahlfeld vorhandenen Textfelder eingetragen. Das linke Feld, in dem der Benutzer einen Wert eintragen kann, hat eine Controlsource aus der Bearbeitungstabelle des Formulars. Das rechte Feld hat eine Controlsource aus der Auswahltabelle und dient der Anzeige von zusätzlichen Werten aus der Auswahltabelle. In der Datenumgebung des Formulars muss eine Relation von der Bearbeitungstabelle zur Auswahltabelle bestehen um die Werte in diesem Textfeld anzuzeigen.

Schließlich werden die Namen der Felder aus der Auswahltabelle benötigt, die nach der Auswahl im Auswahlfeld angezeigt werden sollen. „Return Field Name (Code)“ enthält den Namen des Auswahlfeldes für das linke Textfeld. „Return Field Name (Description)“ enthält den Namen des Auswahlfeldes für das rechte Textfeld.

Zur Laufzeit stehen in der Auswahlliste alle von VFX Grids bekannten Eigenschaften, wie inkrementelle Suche, zur Verfügung. Alle Einstellungen der Auswahlliste werden benutzerspezifisch gespeichert.

Formulare basierend auf der Klasse cDataFormPage

Die mit den VFX – Form Buildern erstellten Formulare haben standardmäßig viele gute Eigenschaften. Die Position des Formulars auf dem Bildschirm, die Größe des Formulars (die Größe eines Formulars kann mithilfe eines Resizers vom Benutzer zur Laufzeit eingestellt werden), die zuletzt aktive Seite des Seitenrahmens sowie die Einstellungen des Grid, Sortierfolge, Spaltenbreiten, werden für jeden Benutzer individuell gespeichert. Schließt ein Benutzer ein Formular und öffnet er es wieder, erscheint es genauso, wie er es verlassen hat.

The screenshot shows a software window titled "Artikel" with two tabs: "Dateneingabe" (Data Entry) and "Liste" (List). The "Dateneingabe" tab is active and contains the following fields:

Originalname:	Chang	Lieferant:	Exotic Liquids
Verkaufsname:	Tibetanisches Chang-Bier	Kategorie:	Getränke
Anzahl pro ME:	24 - 330 ml Flaschen	Mindestbestand:	25,000
Einzelpreis:	14,0000	Bestellt:	40,000
Kosten pro ME:	13,3000	Lagerbestand:	17,000
Auslaufartikel:	<input type="checkbox"/>		

In allen Spalten eines Grid ist standardmäßig eine inkrementelle Suche möglich. Durch einen Doppelklick auf eine Überschrift in einem Grid kann die entsprechende Spalte sortiert werden. Wenn für die Spalte kein geeigneter Index vorhanden ist, wird von VFX automatisch ein temporärer Index angelegt. Soll die Suche um eine zusätzliche Spalte erweitert werden, drückt man die Taste „Strg“ und klickt gleichzeitig auf eine weitere Überschrift. Die Rangfolge der Sortierung wird in den Überschriften durch Zahlen in Klammern dargestellt.

Dateneingabe		Liste		
Originalname	Verkaufsname	Kosten pro ME	Einzelpreis	Verpackungse
Chang	Tibetanisches Ch	13,3000	14,0000	24 - 330 ml Fl
Chef Anton's Caju	Chef Anton's Caju	15,4000	22,0000	48 - 170 ml Gl
Guaraná Fantástico	Guaraná Fantástico	3,1500	4,5000	12 - 355 ml Dc
Sasquatch Ale	Sasquatch Ale	9,8000	14,0000	24 - 330 ml Fl
Steeleye Stout	Steeleye Stout	12,6000	18,0000	24 - 330 ml Fl
Côte de Blaye	Côte de Blaye (rot	184,4500	263,5000	12 - 750 ml Fl
Chartreuse verte	Grüner Chartreus	12,6000	18,0000	750 ml pro Fla
Ipoh Coffee	Malaisischer Kaffee	32,2000	46,0000	16 - 500 g Dos
Laughing Lumber	Laughing Lumber	9,8000	14,0000	24 - 330 ml Fl
Outback Lager	Outback Lager-Bie	10,5000	15,0000	24 - 355 ml Fl
Rhönbräu Kloster	Rhönbräu Kloster	5,4250	7,7500	24 - 0.5 l Flas

Formulare basierend auf der Klasse CTableForm

Bei Formularen basierend auf der Klasse CTableForm sind das Such-Grid und andere Steuerelemente nebeneinander oder untereinander auf einem Container angeordnet. Ein typisches CTableForm-Formular ist die Verwaltung der Benutzerrechte.

Formulare basierend auf der Klasse COneToMany

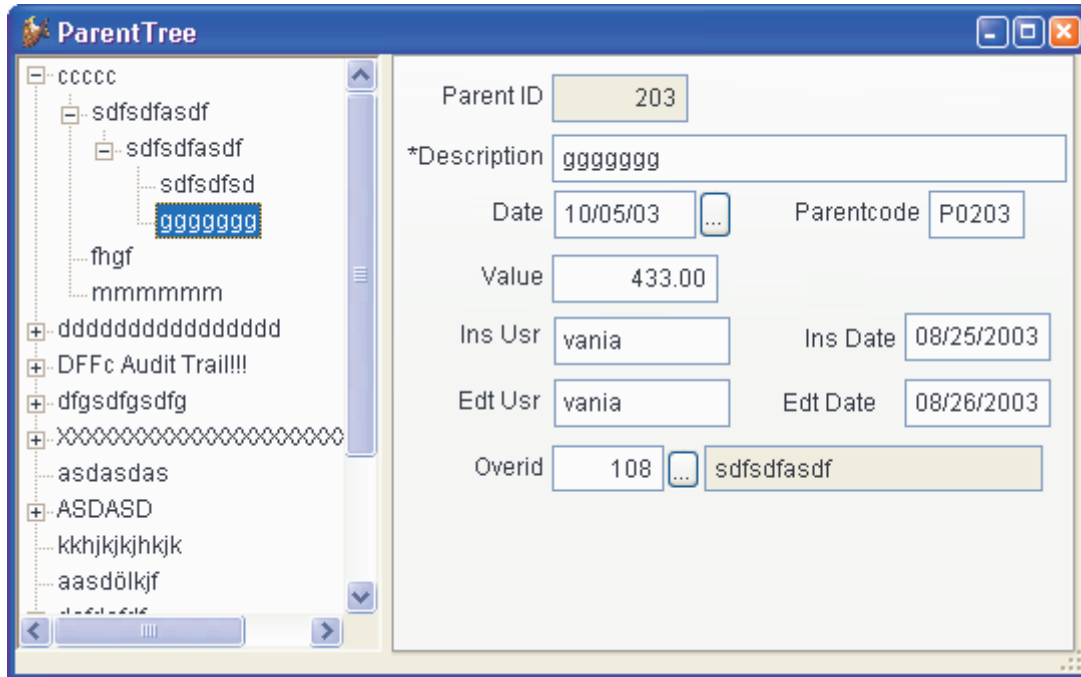
Formulare basierend auf der Klasse COneToMany sehen im oberen Teil wie ein Formular basierend auf CDataFormPage aus. Darunter befindet sich ein zusätzlicher Pageframe mit standardmäßig einer Seite, auf der sich ein Grid zur Anzeige und Bearbeitung von Child-Daten befindet.

Artikel	Menge	Einzelpreis	Gesamtpreis
Boston Crab Meat	998,000	18,4000	18363,2000
Spegesild	24,000	38,5500	925,2000
Valkoinen suklaa	10,000	33,2500	332,5000

Im Gegensatz zu anderen VFX-Formularklassen ist es hier möglich Daten in einem Grid zu bearbeiten. Über die Schaltflächen am unteren Formularrand kann der Benutzer neue Child-Datensätze anfügen oder vorhandene Child-Datensätze löschen.

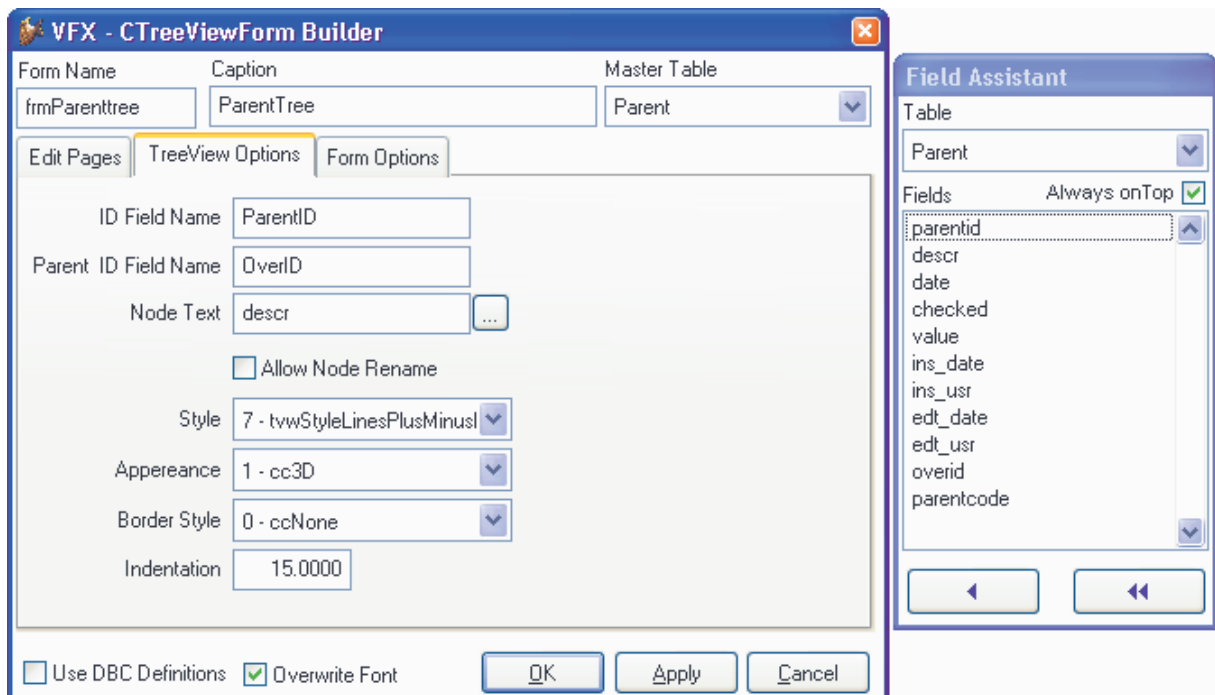
Formulare basierend auf der Klasse cTreeViewForm

Der Haupteinsatzzweck dieser Klasse ist die Darstellung von Daten aus einer Tabelle in einer Baumstruktur. Die Baumstruktur gibt dem Endanwender einen kompletten Überblick über die hierarchischen Beziehungen in einer Tabelle.



Diese Klasse basiert auf der Klasse cDataFormPage (Vfxform.vcx) und enthält ein Treeview-Steuerelement aus der Klasse cTreeView (Vfxappl.vcx). Die Klasse kombiniert die Funktionalität von cDataFormpage mit den Möglichkeiten der hierarchischen Datenpräsentation in einer Baumstruktur. Wenn ein Eintrag im Treeview-Steuerelement ausgewählt wird, wird der Datensatzzeiger in der zugrunde liegenden Tabelle mitgeführt und der Anwender kann die Daten im rechten Teil des Formulars bearbeiten.

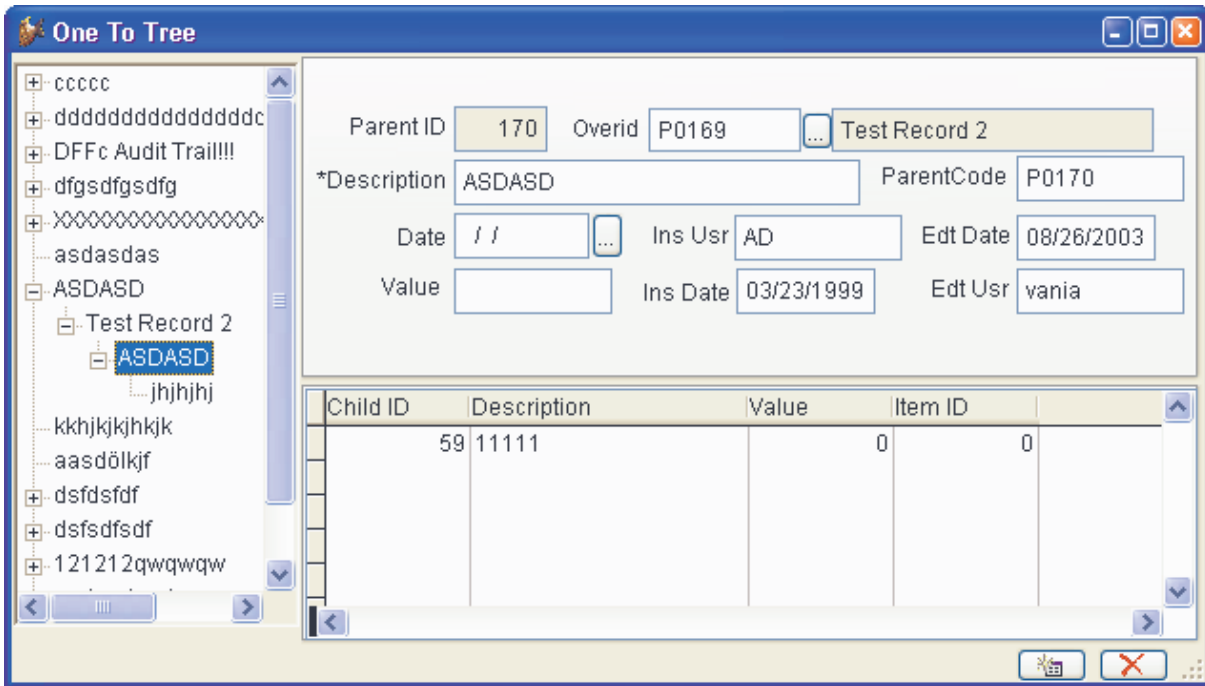
Mit dem VFX – CTreeViewForm Builder können sehr schnell Formulare basierend auf der Klasse cTreeViewForm erstellt und alle benötigten Eigenschaften können eingestellt werden.



Der Builder arbeitet so ähnlich wie der VFX – CDataFormPage Builder. Die Einstellungen können auf den Seiten *Edit Pages* und *Form Options* genauso gemacht werden, wie im VFX – CDataFormPage Builder. Zusätzlich müssen die Einstellungen für das Treeview-Steuerelement auf der Seite *TreeViewOptions* gemacht werden. Es müssen zwei Arten von Einstellungen für das Treeview-Steuerelement gemacht werden.

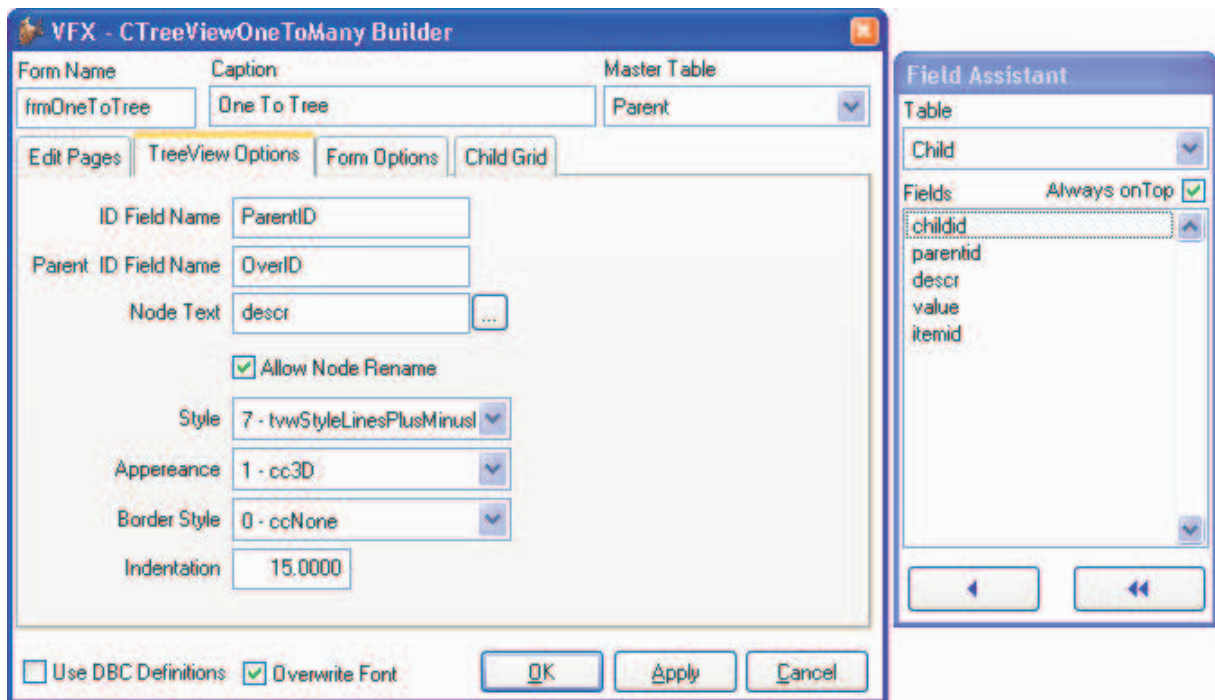
Die Klasse cTreeViewOneToMany

Der Haupteinsatzzweck dieser Klasse ist die Darstellung der Daten aus einer Tabelle in einer Baumstruktur zusammen mit der leistungsfähigen Funktionalität, die die cOneToMany-Klasse dem Entwickler bietet. Die Baumstruktur gibt dem Anwender den kompletten Überblick über die hierarchischen Datenbeziehungen.



Diese Klasse basiert auf der Klasse cOneToMany (Vfxform.vcx) und enthält ein Treeview-Steuerelement aus der Klasse cTreeView (Vfxappl.vcx). Die Klasse kombiniert die Funktionalität von cOneToMany mit den Möglichkeiten der hierarchischen Datenpräsentation in einer Baumstruktur. Wenn ein Eintrag im Treeview-Steuerelement ausgewählt wird, wird der Datensatzzeiger in der zugrunde liegenden Tabelle mitgeführt und der Anwender kann die Daten im rechten Teil des Formulars bearbeiten. Zusätzlich können die Child-Daten im unteren Teil des Formulars bearbeitet werden.

Mit dem VFX – CTreeViewOneToMany Builder können sehr schnell Formulare basierend auf der Klasse cTreeViewOneToMany erstellt und alle benötigten Eigenschaften eingestellt werden.



Dieser Builder arbeitet so ähnlich wie der VFX – COneToMany Builder. Die Einstellungen auf den Seiten *Edit Pages*, *Form Options* und *Child Grid* werden genauso gemacht, wie bei Formularen basierend auf der Klasse *cOneToMany*. Zusätzlich müssen die Einstellungen für das Treeview-Steuerelement auf der Seite *TreeViewOptions* gemacht werden.

Die Einstellungen erfolgen genauso wie beim VFX –CTreeViewForm Builder.

Linked Child-Formulare

Sehr leistungsfähig ist die Möglichkeit Formulare mit dem VFX Child Form Manager zu verbinden. Ein Formular dient dabei als Parent-Formular und eins als Child-Formular. Das Parent-Formular steuert dabei das Child-Formular. Im Child-Formular werden die zum aktuellen Parent-Datensatz gehörigen Datensätze angezeigt.

Der VFX-Entwickler muss dazu im Child-Formular mit dem Form Builder auf der Seite Optionen „Is Child Form“ auswählen oder manuell die Formulareigenschaft *lchildform* auf *.T.* zu setzen.

Beim Parent-Formular müssen mit dem Form Builder die Optionen „Has More Options“ (setzt die Eigenschaft *lmore* auf *.T.*), „Has Child Form“ und „Auto Sync Child Form“ (setzt die Eigenschaft *lautosynchildform* auf *.T.*) ausgewählt werden. Der Form Builder trägt automatisch Template-Code in die Methoden *onmore* und *onsetchilddata* ein. Der Code dieser Methoden muss anschließend manuell bearbeitet werden.

In der Methode *onmore* wird das Child-Formular aufgerufen. Hier ein Ausschnitt aus der Methode *onmore*:

```
cCalledBy      = "PARENT"
lcFixFieldValue = STR(parent.parentid, 10)
lcCaption      = "Child records for parent " + trim(parent.descr)
lcFixFieldName = "parentid"
lcFilterExpr   = "parentid="+str(parent.parentid, 10)

local laFunct[1,5]

laFunct[1,1] = "Child Records"
laFunct[1,2] = "Child Records for selected parent"
laFunct[1,3] = "F"      && W - Wait Window, F - Form to run
laFunct[1,4] = "CHILD"
laFunct[1,5] = lcCalledBy      + ";" +;
               lcFixFieldValue + ";" +;
               lcCaption      + ";" +;
               lcFixFieldName + ";" +;
               lcFilterExpr
```

Der Variablen *cCalledBy* ist der Name des aktuellen Formulars zuzuweisen. Die Variable *lcFixFieldValue* bekommt den Wert des Primärschlüssels des aktuellen Datensatzes. Wenn im Child-Formular ein neuer Datensatz angelegt wird, wird dieser Wert als Fremdschlüssel eingetragen. Die Variable *lcCaption* bekommt die Caption des Child-Formulars zugewiesen. Es ist empfehlenswert hier den Bezug zum Parent-Datensatz einzutragen. So ist es für den Anwender zur Laufzeit einfach ersichtlich zu welchem Parent-Datensatz die Datensätze im Child-Formular gehören. *lcFixFieldName* ist der Name des Feldes mit dem Fremdschlüssel der Child-Tabelle. Wenn neue Child-Datensätze erfasst werden, wird in dieses Feld der Fremdschlüssel eingetragen.

Der Variablen *lcFilterExpr* wird schließlich eine Filterbedingung zugewiesen. Der Filterausdruck muss so formuliert werden, dass damit im Child-Formular die Datensätze passend zum aktuellen Parent-Datensatz angezeigt werden. *lcFilterExpr* wird im Child-Formular als Makro evaluiert.

Das Array *laFunct* enthält die Werte zur Anzeige des OnMore-Dialogs. Die erste Spalte wird in der Listbox angezeigt. Die zweite Spalte enthält eine detailliertere Beschreibung und wird in der Editbox unterhalb der Listbox angezeigt. In der dritten Spalte wird „F“ eingetragen. Dadurch wird VFX kenntlich gemacht, dass ein Formular gestartet werden soll. Der Name des Formulars wird in der vierten Spalte übergeben. In der fünften Spalte werden die Werte der oben definierten Variablen in einer durch Semikolon separierten Liste übergeben.

Nicht dokumentiert ist die Möglichkeit in der dritten Spalte ein „M“ einzutragen. Hiermit kann eine Formularymethode aufgerufen werden.

Die Methode *onsetchilddata* wird aufgerufen, wenn der Satzzeiger bewegt wird. In diesem Fall muss die Anzeige im Child-Formular so aktualisiert werden, dass wieder die zum Parent-Datensatz passenden Child-Datensätze angezeigt werden. Hier der Code:

```
lparameters tcFormName, toForm

*!* ATTENTION: tcFormName is always in UPPERCASE
tcFormName = upper(tcFormName)

do case
  case tcFormName=="FRMCHILD" && 'FRM'+SCX Name
    toForm.Caption      = "Child Records for " + trim(parent.descr)
    toForm.cFixFieldValue = str(parent.parentid)
    toForm.cFilterExpr  = "PARENTID=" + str(parent.parentid, 10)
  endcase
```

Das Objekt *toForm* ist hierbei eine Referenz auf das Child-Formular. Im Prinzip können hier also beliebige Eigenschaften des Child-Formulars gesetzt werden oder auch Methoden aufgerufen werden. Damit das Linked-Child-Szenario funktioniert, müssen drei Eigenschaften gesetzt werden. Der Caption des Child-Formulars wird ein Wert zugewiesen, der auf den Bezug zum Parent-Datensatz schließen lässt. Der Eigenschaft *cFixFieldValue* wird der Wert des Primärschlüssels im Parent-Formular zugewiesen. Schließlich muss der Filter neu gesetzt

werden. Dazu wird der benötigte Ausdruck der Eigenschaft *cFilterExpr* zugewiesen. In der Regel können die benötigten Werte aus der Methode *onmore* kopiert werden.

Ein Parent-Formular kann beliebig viele Child-Formulare steuern. Ein Child-Formular kann wiederum ebenfalls eigene Child-Formulare steuern. Eine komplexe Verkettung von Formularen ist möglich.

Symbolleisten zu Formularen

Es hat sich als sehr praktisch erwiesen Formularen eigene Symbolleisten zuordnen zu können. Die Symbolleisten sollten auf der Klasse *ctoolbar* basieren und in der Klassenbibliothek *Appl.vcx* gespeichert werden. Der Name der Symbolleiste wird dem Formular in der Eigenschaft *ctoolbarclass* bekannt gemacht.

VFX instanziiert die Symbolleiste zusammen mit dem Formular. Die Symbolleiste ist sichtbar, solange das Formular das aktive Formular ist.

Um zum Beispiel ein Child-Formular über eine Schaltfläche in einer Symbolleiste zu öffnen, fügen wir der Symbolleiste eine Schaltfläche basierend auf der Klasse *ctoolbarclass* hinzu. In das Click-Event der Schaltfläche schreiben wir

```
_screen.activeform.onmore(1)
```

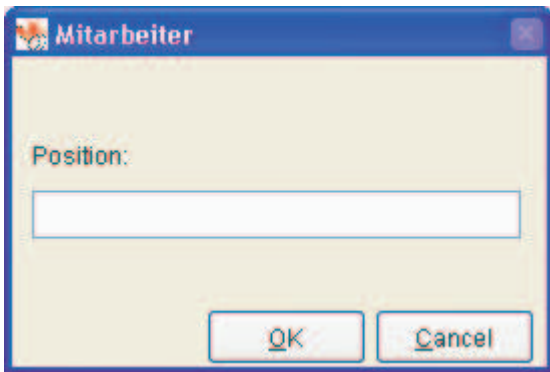
Das ist alles. Da VFX sicherstellt, dass die Symbolleiste nur dann sichtbar ist, wenn das dazugehörige Formular aktiv ist, können wir sicher sein, dass *_screen.activeform* existiert. Von diesem Formular wird die *onmore* Methode aufgerufen und bekommt als Parameter eine 1 übergeben. Damit wird das Formular aufgerufen, das im ersten Array-Element der *onmore* Methode angegeben ist, ohne dass der OnMore-Dialog angezeigt wird.

Formulare basierend auf Ansichten

VFX unterstützt die tabellenorientierte Datenbearbeitung genauso wie die Arbeit mit Ansichten. Wenn die Datenquelle eines Formulars eine Ansicht sein soll, muss auf der Seite Optionen des VF Form Builder das Häkchen bei „Work On View“ gesetzt werden. Damit weiß VFX, dass es sich bei der Datenquelle um eine Ansicht handelt. Ansichten können insbesondere keine Indexschlüssel haben. VFX muss also in jedem Fall, in dem eine Sortierung benötigt wird, eine temporäre Indexdatei erstellen.

In den meisten Fällen sind Ansichten parametrisiert. Die Parameter müssen vor Abfrage der Daten der Ansicht bekannt sein. Zur Eingabe der Ansichtparameter stellt VFX die Formulkategorie *CAskViewArg* zur Verfügung. Das Datenbearbeitungsformular wird wie gewohnt mit dem VFX Form Builder erstellt. Die Eigenschaft *lworkonview* wird auf *.T.* gesetzt. Bei der Ansicht in der Datenumgebung wird die Eigenschaft *nodataonload* auf *.T.* gesetzt. Das bedeutet, dass die Ansicht beim Laden des Formulars geöffnet wird, ohne dass Datenabgefragt werden.

Jetzt wird ein neues Formular basierend auf der Klasse *CAskViewArg* erstellt. Die Steuerelemente, die als *ControlSource* Felder enthalten, die auch als Ansichtparameter verwendet werden, können über die Zwischenablage vom Bearbeitungsformular auf das Formular basierend auf der Klasse *CAskViewArg* kopiert werden. In der Eigenschaft *cviewparameter* ist der Name des Ansichtparameters einzutragen. Den Steuerelementen können geeignete Bezeichnungen hinzugefügt werden. Das Formular ist damit fertig und kann gespeichert werden.



Aus dem Bearbeitungsformular muss nun noch das Formular basierend auf der Klasse `CAskViewArg` aufgerufen werden. Dies geschieht am Ende des Init-Events:

```
do form <Formular zur Eingabe der Ansichtsparameter> with this
```

Es ist auch möglich zur Laufzeit des Formulars das Formular zur Eingabe der Ansichtsparameter erneut aufzurufen. Wenn der Aufruf aus einem Steuerelement, zum Beispiel aus dem Click Event einer Schaltfläche erfolgt, muss der Aufruf so aussehen:

```
do form <Formular zur Eingabe der Ansichtsparameter> with thisform
```

Mehr ist bei der Arbeit mit Ansichten nicht zu beachten. Alles Weitere erledigt VFX.

Applikationsschutz durch Produktaktivierung

Das Ziel der Produktaktivierung ist die unerlaubte Verwendung der Anwendung auf nicht aktivierten Computern zu verhindern.

Der Applikationsschutz durch Produktaktivierung kann im VFX – Application Wizard auf der Seite 3. *Options* durch aktivieren des Kontrollkästchens „*Enable product activation*“ für ein neu zu erstellendes Projekt eingeschaltet werden.

Später kann diese Einstellung in `Vfxmain.prg` geändert werden. Die Eigenschaft `goProgramm.IUseActivation` muss auf `.T.` gesetzt werden, um die Produktaktivierung einzuschalten. Wenn die Eigenschaft `goProgramm.IUseActivation` auf `.F.` gesetzt ist, ist die Applikation nicht durch die Produktaktivierung geschützt.

Zu jeder Anwendung können bis zu 32 Rechte vergeben werden. Jedes Recht kann unabhängig von den anderen Rechten aktiviert werden.

Liste der verwendeten Begriffe

Systemspezifischer Wert – Ein systemspezifischer Wert, zum Beispiel die Seriennummer einer Hardware-Komponente oder das Erstellungsdatum einer bestimmten Datei oder ein Schlüssel aus der Windows-Registrierungsdatenbank. Die zu verwendete Datei und der zu verwendende Schlüssel aus der Windows-Registrierungsdatenbank können vom Entwickler festgelegt werden.

Aktivierungsregel – Für jede Applikation kann eine eindeutige Aktivierungsregel angelegt werden. Diese Regel setzt sich aus einer Reihe systemspezifischer Werte zusammen, die einen PC eindeutig identifizieren. Bei der Erstellung der Aktivierungsregel können Textbearbeitungsfunktionen verwendet werden.

Installationsschlüssel – Dies ist eine Zeichenkette, die Informationen über die im PC des Anwenders eingesezte Hardware enthält. Der Installationsschlüssel wird vom Entwickler benötigt um einen Aktivierungsschlüssel erstellen zu können.

Aktivierungsschlüssel – Dies ist eine Zeichenkette, die die Berechtigungen für einen speziellen PC enthält. Der Aktivierungsschlüssel wird vom Entwickler anhand des Installationsschlüssels erstellt. Der Aktivierungsschlüssel ist für andere PCs nutzlos.

Installationsdatum – An diesem Datum wurde eine Applikation erstmalig auf einem PC gestartet.

Das Funktionsprinzip

Wenn der Applikationsschutz durch Produktaktivierung aktiviert ist, wird beim Start der Applikation das Objekt *goProgram.SecurityRights* instanziiert. Dieses Objekt hat Eigenschaften mit den Namen der Benutzerrechte, die der Entwickler definiert hat. Jede dieser Eigenschaften kann einen von drei Werten annehmen:

- 1 – Die Applikation ist nicht aktiviert. In diesem Fall kann der Entwickler entscheiden welche Aktion ausgeführt werden soll. Der Anwender könnte zum Beispiel begrenzten Zugriff auf Funktionen haben, solange die Applikation nicht aktiviert ist.
- 0 – Die Applikation ist aktiviert, aber der Anwender hat nicht das Recht diese Aktion auszuführen.
- 1 – Die Applikation ist aktiviert und der Anwender hat das Recht die Aktion auszuführen.

Wenn der Applikationsschutz durch Produktaktivierung aktiviert ist, werden der Aktivierungsschlüssel und das Datum des ersten Starts der Anwendung in einer Ini-Datei gespeichert. Der Entwickler kann den Namen dieser Ini-Datei selbst wählen, sodass jede Applikation ihre eigene Ini-Datei verwendet. Der Standardname ist *VFX.ini*. Die Ini-Datei wird im Windows-Ordner gespeichert.

Der Aktivierungsschlüssel wird durch die Aktivierungsregel verschlüsselt. Der Schutz kann durch Hinzufügen von Zeichenkonstanten, Schlüsseln aus der Windows-Registrierungsdatenbank und durch das Erstellungsdatum einer beliebigen Datei weiter verbessert werden. Diese Kombination kann für jede Applikation getrennt festgelegt werden, sodass jede Applikation ihre eigenen Aktivierungsregeln hat.

Zusätzlich zu diesen Einstellungen kann der Entwickler den Typ des Schutzes festlegen.

Der Standardschutz erstellt die Ini-Datei beim ersten Start der Applikation. Das während des Erstellens der Ini-Datei aktuelle Systemdatum wird in der Datei gespeichert. Dieses Datum steht während der Ausführung der Anwendung in der Eigenschaft *goProgram.InstallationDate* zur Verfügung und kann dazu verwendet werden die Laufzeit der Applikation zu beschränken. Der Nachteil dieses Schutzes ist, dass der Anwender die erstellte Ini-Datei löschen kann und die Ini-Datei beim nächsten Start der Applikation mit einem neuen Datum erneut erstellt wird.

Um eine solche Manipulation durch den Anwender auszuschließen, kann der Entwickler einen erweiterten Schutz einstellen. Hierbei wird eine zusätzliche Datei verwendet, die mit der Applikation vertrieben werden muss. Der Standardname dieser Datei heißt „FirstInstall.txt“. Der Dateiname kann mit der Eigenschaft *cFirstInstall* aus der Klasse *cActivation* (appl.vcx) eingestellt werden. Die Datei „FirstInstall.txt“ wird im Windows-Ordner abgelegt.

Wenn der Entwickler den Schutz mit der Datei „FirstInstall.txt“ auswählt, wird sich die Applikation folgendermaßen verhalten. Beim Start der Applikation wird zunächst die Ini-Datei überprüft. Wenn diese Datei existiert, wird das Datum des ersten Starts der Eigenschaft *goProgram.InstallationDate* zugewiesen und die Benutzerrechte werden entsprechend dem Aktivierungsschlüssel eingestellt.

Wenn die Ini-Datei nicht existiert wird angenommen, dass dies der erste Start der Applikation ist. Wenn dies der Fall ist, wird zusätzlich überprüft, ob die Datei „FirstInstall.txt“ existiert. Wenn diese Datei existiert ist sichergestellt, dass die Applikation wirklich zum ersten Mal gestartet wird. Das Systemdatum wird jetzt in der Ini-Datei gespeichert und die Datei „FirstInstall.txt“ wird gelöscht. Wenn ein Anwender nun versucht eine Applikation zu reaktivieren indem er die Ini-Datei löscht, wird die Ausführung der Applikation beendet, weil die Datei „FirstInstall.txt“ nicht existiert. Dieser erweiterte Schutz der Applikation bedeutet eine bessere Sicherheit. Der Entwickler darf jedoch nicht vergessen die Datei „FirstInstall.txt“ beim Vertrieb der Applikation mit auszuliefern.

Wenn der Anwender die installierte Applikation aktivieren möchte, muss er seinen Installationsschlüssel an den Entwickler senden. Der Installationsschlüssel kann auf drei verschiedene Arten an den Entwickler gesendet werden. Die gewünschte Art kann in der Eigenschaft *nRegWay* eingestellt werden:

- 0 – Der Installationsschlüssel wird in einem Dialog angezeigt. Der Anwender kann den Schlüssel kopieren und in einer anderen Applikation (zum Beispiel in einer E-Mail) einfügen.
- 1 – Der Installationsschlüssel wird in einer Datei gespeichert. Diese Datei kann später an den Entwickler gesendet werden. Der Dateiname wird in der Eigenschaft *cParamFile* hinterlegt.

2 – Der Installationsschlüssel wird in einer Datei gespeichert und sofort als E-Mail-Anhang an den Entwickler geschickt. Der Dateiname muss in der Eigenschaft *cParamFile* hinterlegt werden. Die E-Mail-Adresse des Entwicklers muss in der Eigenschaft *cRegEMail* eingetragen werden.

Der Installationsschlüssel hat einen numerischen Wert mit 10 Stellen Länge. Der Anwender könnte den Installationsschlüssel per E-Mail an den Entwickler senden oder auf einer Registrierungs-Website eintragen. Der Entwickler trägt den Installationsschlüssel im „Create Activation Key“-Assistenten ein um einen Aktivierungsschlüssel für den Anwender zu erstellen. Der generierte Aktivierungsschlüssel wird dann an den Anwender geschickt und vom Anwender im Aktivierungsformular eingegeben um die Applikation zu aktivieren. Wahlweise kann die Datei mit dem Aktivierungsschlüssel auch einfach im Ordner der Exe-Datei gespeichert werden. Beim nächsten Start der Anwendung wird der Aktivierungsschlüssel aus dieser Datei gelesen.

Die Aktivierungsinformationen werden auf dem PC des Kunden in einer Ini-Datei gespeichert. Der Name dieser INI-Datei wird in der Eigenschaft *cINIFileName* der Klasse *cVFXActivation* (Appl.vcx) eingetragen. Der Standardwert ist *VFX.ini*.

Der Entwickler kann wählen, ob die einfache Produktaktivierung verwendet werden soll oder ob zusätzlich die Datei „FirstInstall.txt“ benutzt werden soll, um den ersten Start der Applikation zu protokollieren. Der Name dieser Datei kann in der Eigenschaft *cFirstInstall* der Klasse *cVFXActivation* (Appl.vcx) eingetragen werden. Der Standardwert ist *FirstInstall.ini*.

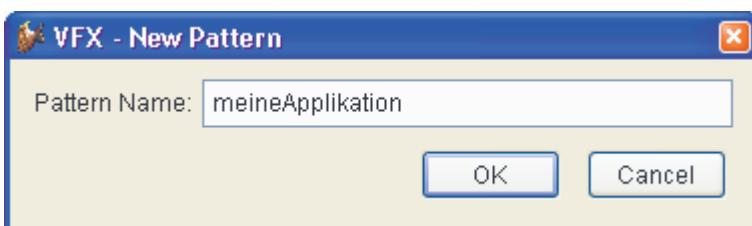
Wenn die Datei „FirstInstall.txt“ verwendet werden soll, muss diese Datei mit der Applikation vertrieben werden. Das Installationsprogramm muss diese Datei im Windows-Ordner speichern. Das Aktivierungsobjekt wird diese Datei beim ersten Start der Applikation löschen. In diesem Moment wird das Installationsdatum in der INI-Datei gespeichert. Später wird bei jedem Start der Applikation in der INI-Datei geprüft, ob das Installationsdatum vorhanden ist. Wenn das Datum fehlt und wenn die Datei „FirstInstall.txt“ nicht vorhanden ist, wird davon ausgegangen, dass an der Installation manipuliert wurde und die Ausführung der Applikation wird beendet.

Wenn die Datei „FirstInstall.txt“ nicht verwendet wird, wird die Ini-Datei neu erstellt, falls sie nicht vorhanden ist.

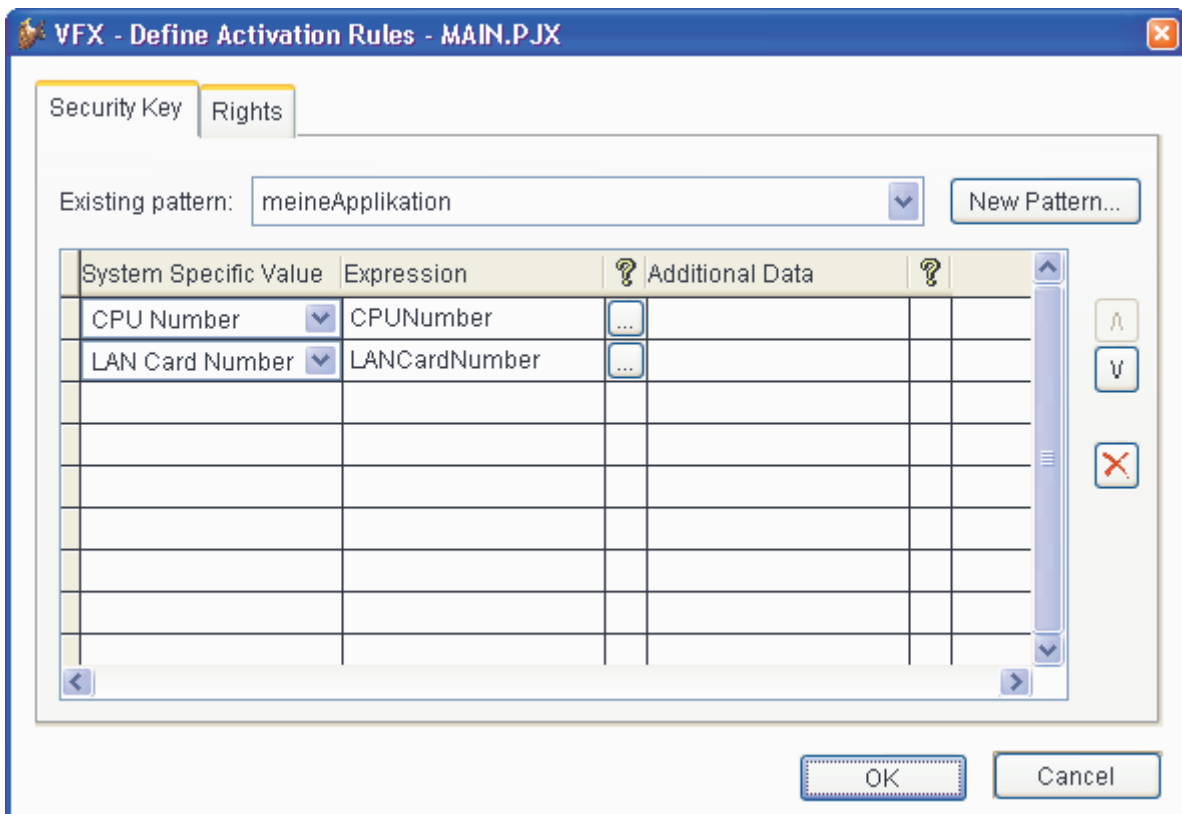
Das Installationsdatum kann auf zwei Arten ermittelt werden. Entweder wird das Systemdatum verwendet oder es wird das Erstellungsdatum einer bestimmten Datei verwendet. Wenn das Erstellungsdatum einer Datei verwendet werden soll, muss der Name dieser Datei in der Eigenschaft *cRegFileName* der Klasse *cVFXActivation* gespeichert werden.

Die Definition der Aktivierungsregeln

Wenn der „Define Activation Rules“-Assistent das erste Mal für ein Projekt gestartet wird, muss eine neue Regel für dieses Projekt angelegt werden.



Nach der Eingabe eines Namens für die Regel wird der „Define Activation Rules“-Assistent gestartet.



Auf der Seite *Security Key* des Assistenten befindet sich eine Combobox aus der eine Regel für das aktuelle Projekt ausgewählt werden kann. In dem darunterliegenden Grid können so viele Zeilen hinzugefügt werden, wie benötigt werden. Aus allen Zeilen des Grids wird in ein Schlüssel generiert, der in der Eigenschaft *cactpattern* der Klasse *cVfxactivation* gespeichert wird. Die Anwendung beim Kunden erkennt anhand dieses Schlüssels welche systemspezifischen Werte des PCs zur Generierung des Installationsschlüssels verwendet werden müssen. Der Installationsschlüssel stellt sicher, dass die Applikation nur auf dem Computer ausgeführt wird, für den der Aktivierungsschlüssel erstellt wurde.

In der ersten Spalte des Grid kann ein systemspezifischer Wert ausgewählt werden. In einer Combobox sind hier alle möglichen Parameter aufgeführt, die zur Erstellung des Installationsschlüssels verwendet werden können. Zusätzlich können Zeichenkettenfunktionen angewendet werden, um den Wert zu verändern.

Zum Beispiel sollen anstelle der vollständigen Seriennummer einer Festplatte nur die letzten vier Stellen zur Erstellung des Installationsschlüssels verwendet werden. Aus der Combobox in der ersten Spalte wird „HDD Factory Serial Number“ ausgewählt. Die VFX-Systemvariable, die diesem Parameter entspricht heißt „HDDFactoryNumber“ und erscheint in der zweiten Spalte. Um nur die letzten vier Stellen zu verwenden, muss der folgende Ausdruck in der zweiten Spalte eingetragen werden: `RIGHT(ALLTRIM(HDDFactoryNumber),4)`.

Wenn einer der systemspezifischen Werte „File Creation Date“ oder „Registry Key Value“ verwendet werden soll, müssen weitere Parameter angegeben werden. Wenn das Erstellungsdatum einer Datei verwendet werden soll, muss der Name der Datei angegeben werden. Um einen Windows-Registrierungsschlüssel verwenden zu können, muss die Bezeichnung des Schlüssels eingegeben werden. Dies geschieht in der Spalte „Additional Data“.

Aus den Aktivierungsregeln wird auf dem PC des Anwenders ein Installationsschlüssel erstellt. Dabei werden alle in den Aktivierungsregeln enthaltenen Parameter berücksichtigt. Wenn nur ein Parameter auf dem PC des Anwenders verändert wird, wird die Installation ungültig und der Anwender muss einen neuen Aktivierungsschlüssel anfordern, entsprechend seiner geänderten Hardware.

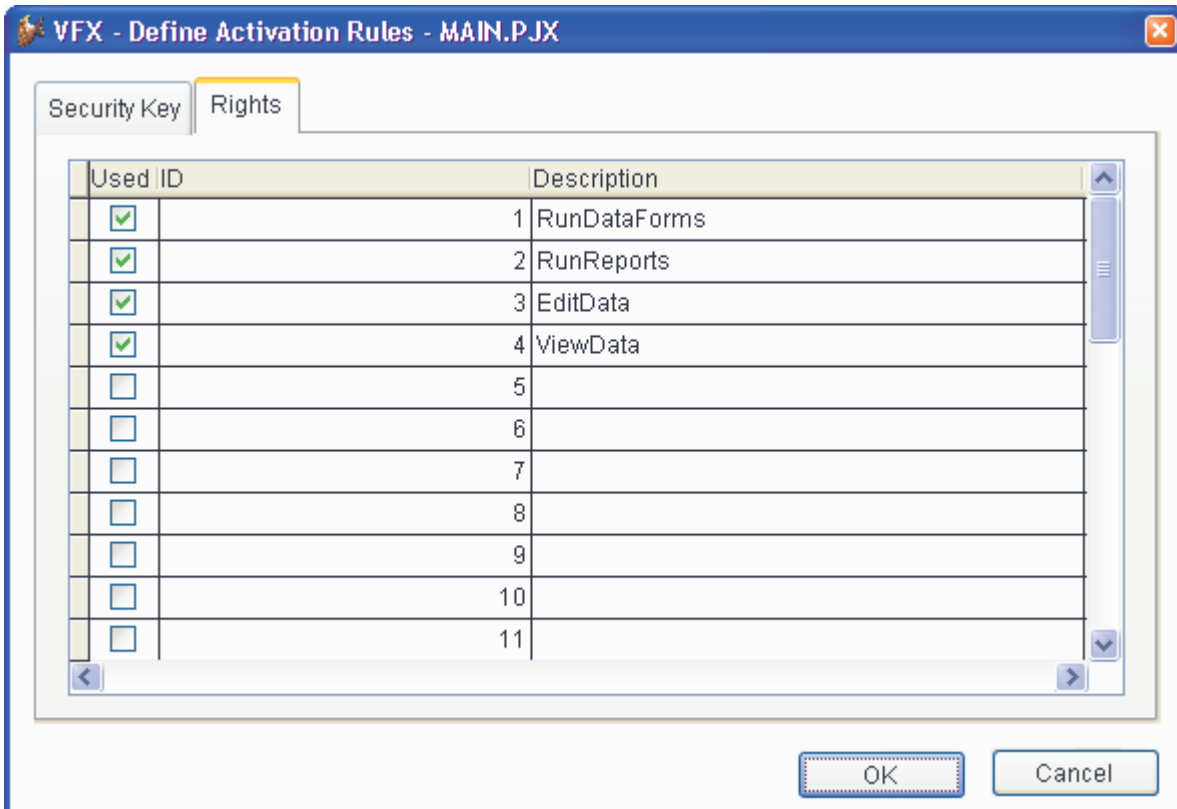
Es können so viele Zeilen dem Grid hinzugefügt werden, wie benötigt werden. Die Zeilen im Grid können mit den Pfeiltasten am rechten Rand des Assistenten in eine andere Reihenfolge gebracht werden. Durch verschieben der Zeilen im Grid ändern sich die Aktivierungsregeln.

Je mehr Zeilen dem Grid hinzugefügt werden, desto länger werden die Aktivierungsschlüssel!

Nach der Definition der Aktivierungsregeln wird das Muster in der Eigenschaft *cActPattern* der Klasse *cVFXActivation* (Appl.vcx) gespeichert.

Achtung: Der Wert der Eigenschaft *cActPattern* darf niemals gelöscht werden! Ohne diesen Wert ist es nicht möglich Aktivierungsschlüssel zu erstellen!

Auf der Seite *Rights* können bis zu 32 verschiedene Benutzerrechte angelegt werden. Damit kann der Zugriff auf bis zu 32 Module einer Applikation gesteuert werden. Beispielsweise können Rechte angelegt werden, die es dem Anwender erlauben Formulare zu starten (*RunDataForms*), Berichte zu drucken (*RunReports*), Daten zu bearbeiten (*EditData*), Daten anzusehen (*ViewData*) usw. Zur Laufzeit der Applikation können die einzelnen Berechtigungen überprüft werden und ggf. wird die entsprechende Aktion ausgeführt.



Alle Benutzerrechte stehen zur Laufzeit als Eigenschaften des global sichtbaren Objekts *goProgram.SecurityRights* zur Verfügung, sodass an jeder Stelle der Applikation darauf zugegriffen werden kann.

Wenn die Applikation nicht aktiviert ist, haben alle Benutzerrechte den Wert -1. Wenn die Applikation aktiviert ist, hat ein Benutzerrecht den Wert 1, wenn die Aktion erlaubt ist und 0, wenn die Aktion nicht erlaubt ist.

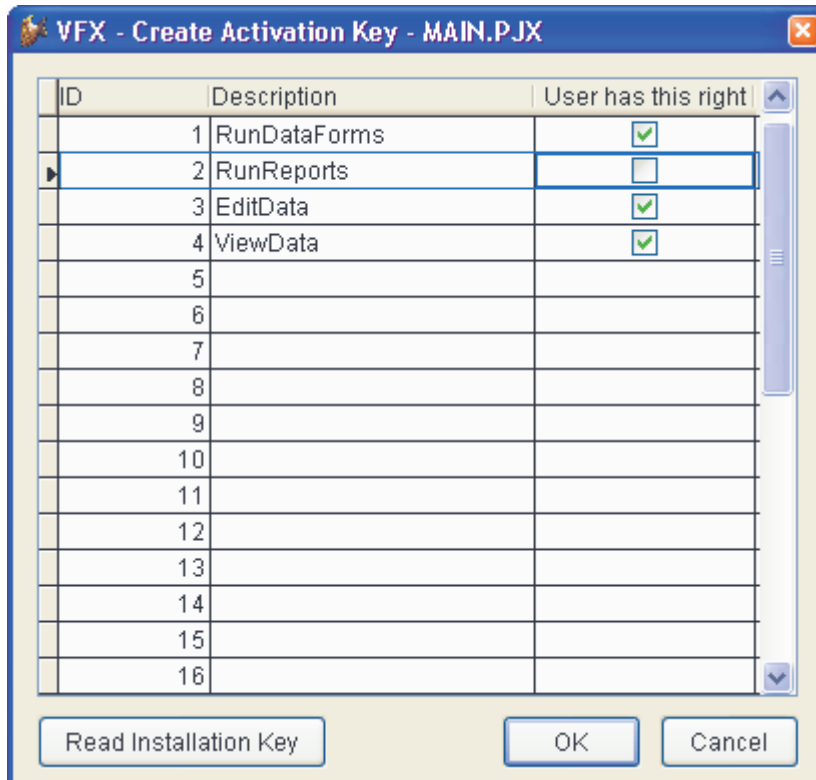
Um im Assistenten ein Recht einzutragen muss zuerst das Kontrollkästchen in der ersten Spalte markiert werden. Dann wird ein Name für das Recht eingetragen. Zur Laufzeit der Applikation wird eine Eigenschaft des *SecurityRights*-Objekts mit diesem Namen angelegt. Daher müssen bei der Eingabe des Namens die Konventionen zur Namensgebung von VFP beachtet werden!

Anmerkung: Applikationsrechte sind für jede Applikation unterschiedlich. Die Rechte, die für eine andere Applikation erstellt wurden, können nicht verwendet werden. Auch wenn ähnliche Rechte benötigt werden, müssen diese neu erstellt werden. Die Applikationsrechte werden in der Tabelle *Vfxapprights.dbf* im Projektordner gespeichert.

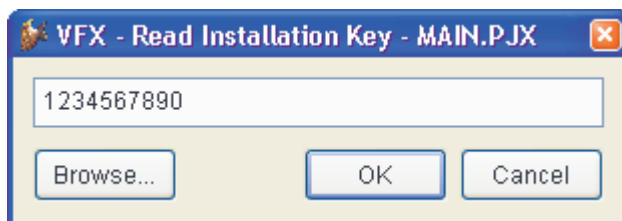
Erstellen eines Aktivierungsschlüssels

Wenn der Anwender seinen Installationsschlüssel sendet, muss ein Aktivierungsschlüssel erstellt werden. Dieser Aktivierungsschlüssel teilt der Applikation mit, ob der Anwender eine bestimmte Aktion ausführen darf. Für jede Aktion muss das entsprechende Recht ausgewählt werden.

Wenn aus dem VFX 8.0-Menü „Create Activation Key“ aufgerufen wird, erscheint der Dialog mit dem Benutzerrechten für das aktive Projekt.



Mit der Schaltfläche „Read Installation Key“ öffnet sich ein Dialog, in den der Installationsschlüssel des Anwenders eingegeben wird. Der Installationsschlüssel kann über die Zwischenablage eingefügt werden oder aus einer Datei gelesen werden.



Nachdem jedes für den Anwender erlaubte Recht markiert ist, wird mit einem Klick auf OK der Aktivierungsschlüssel generiert. Der erstellte Aktivierungsschlüssel wird in der Datei *<Projektname>.xak* im Projektordner gespeichert. Der Aktivierungsschlüssel oder die Datei muss an den Anwender zur Aktivierung der Applikation gesendet werden.

Wenn dem Anwender entsprechend dem obigen Beispiel alle Rechte zur Datenbearbeitung gegeben wurden, er aber nicht das Recht hat Berichte zu drucken, sehen die Eigenschaften zur Laufzeit so aus:

```
goProgram.SecurityRights.RunDataForms = 1
goProgram.SecurityRights.RunReports = 0
goProgram.SecurityRights.EditData = 1
goProgram.SecurityRights.ViewData = 1
```

Wenn der Anwender eine Applikation startet, die eine Aktivierung erfordert (und wenn die Applikation noch nicht aktiviert wurde), wird automatisch der Installationsschlüssel erzeugt. Abhängig vom Wert der Eigenschaft *nRegWay* wird der Installationsschlüssel entweder angezeigt oder in einer Datei gespeichert, die per E-Mail versendet werden kann. Nachdem der Anwender den Aktivierungsschlüssel erhalten hat, kann er ihn im

Aktivierungsfenster eingeben oder die Datei mit dem Aktivierungsschlüssel im Projektordner speichern. Damit ist die Applikation auf diesem Computer aktiviert.

Wenn der Anwender später den Menüpunkt *Hilfe, Produkt aktivieren* auswählt, wird der Installationsschlüssel angezeigt, unabhängig von der Einstellung der Eigenschaft *nRegWay*.

Eigenschaften der Klasse *cVFXActivation*

cFirstInstall – Diese Eigenschaft enthält den Namen einer Datei. Anhand des Vorhandenseins dieser Datei entscheidet diese Klasse, ob die Applikation erstmalig gestartet wird. Wenn dieser Eigenschaft eine leere Zeichenkette zugewiesen wird, kann nicht überprüft werden, ob die Applikation erstmalig gestartet wird. Das Datum des Starts wird dann ohne weitere Überprüfung in der Ini-Datei eingetragen.

cINIFileName – Der Name der Ini-Datei, in der die Aktivierungsinformationen und das Datum des ersten Applikationsstarts gespeichert sind. Der Standardwert ist „VFX.INI“.

cParamFile – Der Name der Datei, in der der Installationsschlüssel gespeichert wird. Abhängig vom Wert der Eigenschaft *nRegWay* kann diese Datei per E-Mail versendet oder auf einem anderen Weg verarbeitet werden.

cRegMail – In dieser Eigenschaft wird die E-Mail-Adresse des Entwicklers gespeichert, an die die Datei mit dem Installationsschlüssel gesendet wird, wenn die Eigenschaft *nRegWay* den Wert 2 hat.

cRegFileName – Hier kann der Name einer Datei angegeben werden, die bei der Installation erstellt wird. Das Erstellungsdatum dieser Datei wird verwendet um das Installationsdatum zu ermitteln. Wenn dieser Eigenschaft kein Wert zugewiesen wird, wird das Systemdatum des ersten Starts der Anwendung verwendet.

nRegWay – In dieser Eigenschaft kann eingestellt werden, wie der Entwickler den Installationsschlüssel bekommen soll.

0 – Der Installationsschlüssel wird in einem Dialog angezeigt und der Anwender kann den Installationsschlüssel kopieren und in beliebige Applikationen einfügen.

1 – Der Installationsschlüssel wird in einer Datei gespeichert. Der Anwender kann diese Datei später an den Entwickler übermitteln. Der Name der Datei wird in der Eigenschaft *cParamFile* hinterlegt.

2 – Der Installationsschlüssel wird in einer Datei gespeichert und an den Entwickler als E-Mail-Anhang gesendet. Der Name der Datei wird in der Eigenschaft *cParamFile* hinterlegt. Die E-Mail-Adresse des Entwicklers, an die der Installationsschlüssel gesendet wird, wird in der Eigenschaft *cRegEMail* eingetragen.

Öffnen-Dialog im Windows-XP-Stil

Zusätzlich zu dem in bisherigen VFX-Versionen vorhandenem Öffnen-Dialog (*Vfxopen.scx*) steht in VFX 8.0 ein neuer Öffnen-Dialog im Windows-XP-Stil (*Vfxpopen.scx*) zur Verfügung. Dieser neue Öffnen-Dialog ist standardmäßig aktiviert. Mit der Eigenschaft *goprogram.lxpopenstyle* kann auf Wunsch auf den alten Öffnen-Dialog umgeschaltet werden.



lxpopenstyle

.T. – der neue Öffnen-Dialog im Windows-XP-Stil wird verwendet.

.F. – der alte Öffnen-Dialog (Vfxfopen.scx) wird verwendet.

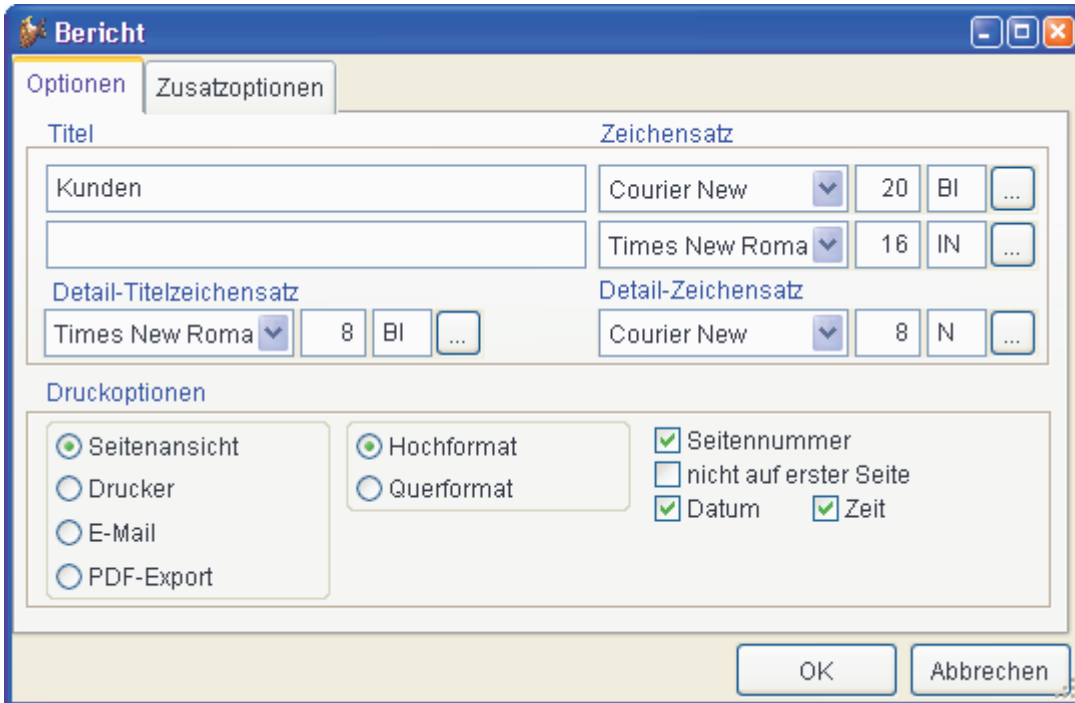
Die Gruppenüberschriften im neuen Öffnen-Dialog werden aus dem neuen Tabellenfeld *Vfxopen.groupcap* gelesen. Der Zustand der einzelnen Gruppen (aufgeklappt oder zugeklappt) wird je Benutzer gespeichert.

Der Datei/Öffnen-Dialog benutzt die Tabelle *VFXFOPEN.DBF*. Die VFX-Formular-Builder fügen automatisch für jedes Formular einen Datensatz zu der Tabelle *VFXFOPEN.DBF* hinzu. Hier ist die Struktur der Tabelle *VFXFOPEN.DBF*:

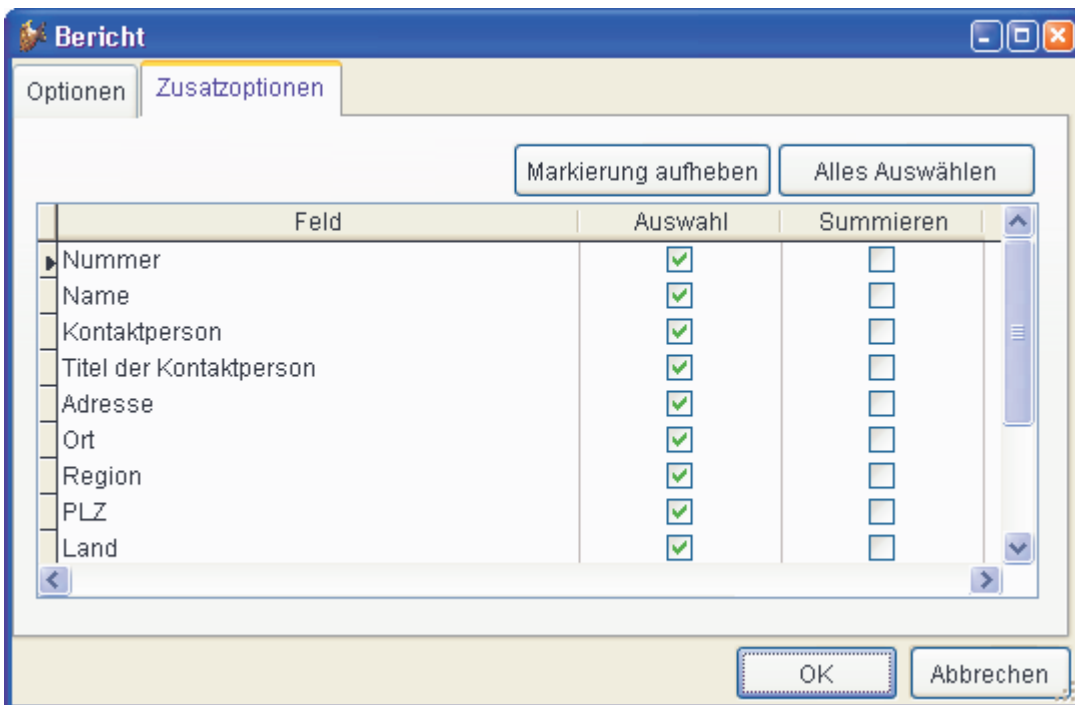
<i>VFXFOpen-Feld</i>	<i>Beschreibung</i>	<i>Beispiel</i>
ObjectID	Diese Feld wird verwendet, wenn der Öffnen-Dialog <i>Vfxfopen.scx</i> verwendet wird. Hierzu muss die Eigenschaft <i>goprogram.lxpopenstyle=.F.</i> gesetzt sein. Der VFX-Öffnen-Dialog hat normalerweise zwei Seiten. (Tip: Sie können die <i>Pagecount</i> -Eigenschaft des Seitenrahmens im Formular <i>Vfxopen.scx</i> auf jeden beliebigen Wert setzen, um die Anzahl der Seiten zu verändern.) Wenn Sie wollen, dass Ihr Formular auf Seite 1 des Seitenrahmens erscheint, geben Sie PAGE1 ein. Für die weiteren Seiten PAGE2, PAGE3 usw.	PAGE1
ObjectNo	Geben eine Zahl für die Sortierfolge der Liste ein. 1 wird das erste Element, es folgt 2 usw. Die Sortierung wird auf jeder Seite benutzt.	1
GroupCap	Diese Feld wird verwendet, wenn der Öffnen-Dialog <i>Vfxpopen.scx</i> verwendet wird. Hierzu muss die Eigenschaft <i>goprogram.lxpopenstyle=.T.</i> gesetzt sein. Dieses Feld enthält eine Gruppenüberschrift. Die Gruppierung erfolgt entsprechend der Einträge im Feld <i>ObjectID</i> . Die <i>GroupCap</i> muss nur für den ersten Eintrag einer Gruppe eingetragen werden.	Kontakte
Title	Geben Sie die Überschrift ein, die im Listenfenster erscheint.	Kunden
Descr	Geben Sie einen Beschreibungstext ein, der angezeigt wird, wenn der Benutzer diesen Eintrag ausgewählt hat.	Liste aller Adressen
Form	Geben Sie den Namen des aufzurufenden Formulars ein.	ADRE
Parameter	Wenn Sie an das Formular Parameter übergeben wollen, können Sie diese hier eingeben.	
Viewlevel	Die Benutzerstufe, die erforderlich ist, um ein Formular anzusehen (Zum Beispiel 1 = Admin, 2 = Hauptbenutzer, 3 = normaler Benutzer usw.)	1 (nur Administratoren können dieses Formular ansehen)
NewLevel	Die Benutzerstufe, die erforderlich ist, um neue Datensätze dem Formular hinzufügen zu können.	1 (nur Administratoren können neue Datensätze hinzufügen)
EditLevel	Die Benutzerstufe, die erforderlich ist, um Datensätze bearbeiten zu können.	1 (nur Administratoren können Datensätze bearbeiten)
Eraselevel	Die Benutzerstufe, die erforderlich ist um auf diesem Formular Datensätze löschen zu können.	1 (nur Administratoren können Datensätze löschen)

Erweiterte Druckoptionen

Aus allen Formularen kann standardmäßig eine Liste gedruckt werden, ohne dass dafür Berichte angelegt werden müssen. VFX legt zur Laufzeit der Anwendung temporäre Berichtsdateien an, die auf der Ansicht der Suchseite eines Formulars basieren.

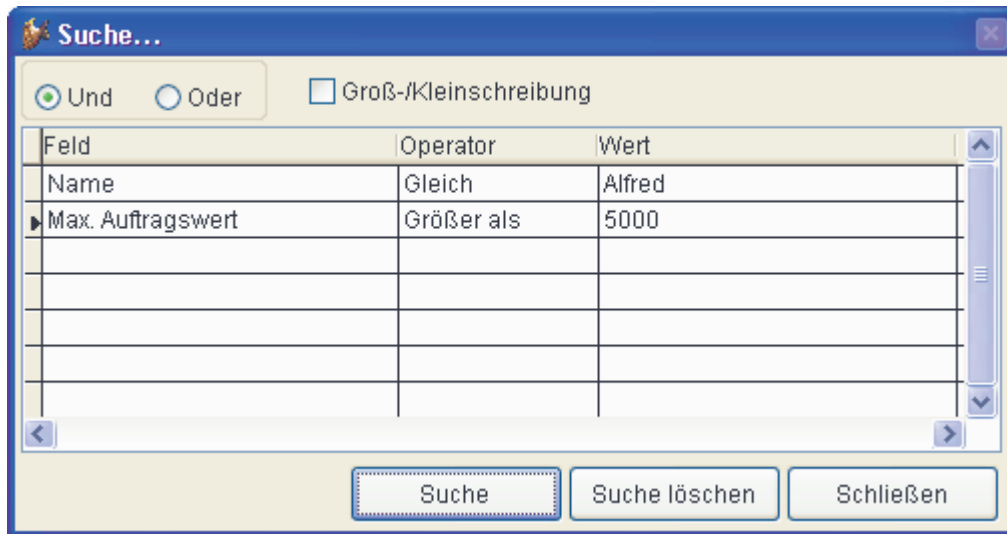


Vor dem Druck bzw. der Seitenansicht kann der Benutzer nicht gewünschte Spalten aus der Liste entfernen. Die Breite der Spalten entspricht ungefähr der Breite der Spalte im Grid.



Suchdialog

Der Suchdialog, der in allen VFX-Formularen zur Verfügung steht, wurde erweitert:



Es können jetzt beliebig viele Suchkriterien kombiniert werden.

Aktualisierung einer SQL Server Kundendatenbank

Der Metadata Wizard hilft Ihnen Metadaten aus Ihrer aktuell benutzten SQL Server-Datenbank zu erstellen. Die Metadaten können zur Aktualisierung der Datenbank beim Kunden verwendet werden.

VFX - Metadata Wizard - MAIN.pjx

Use Database connections

Select SQL Server

Server Name: (local) Use Trusted Connection

User Name:

Password:

Click on next to proceed.

Cancel < Back Next > Finish

Wahlweise kann die Verbindung aus einer VFP-Datenbank ausgelesen werden um die Verbindung zu einem SQL Server herzustellen oder der SQL Server kann manuell ausgewählt werden.

VFX - Metadata Wizard - MAIN.PJX

Database name: test

Connection name:

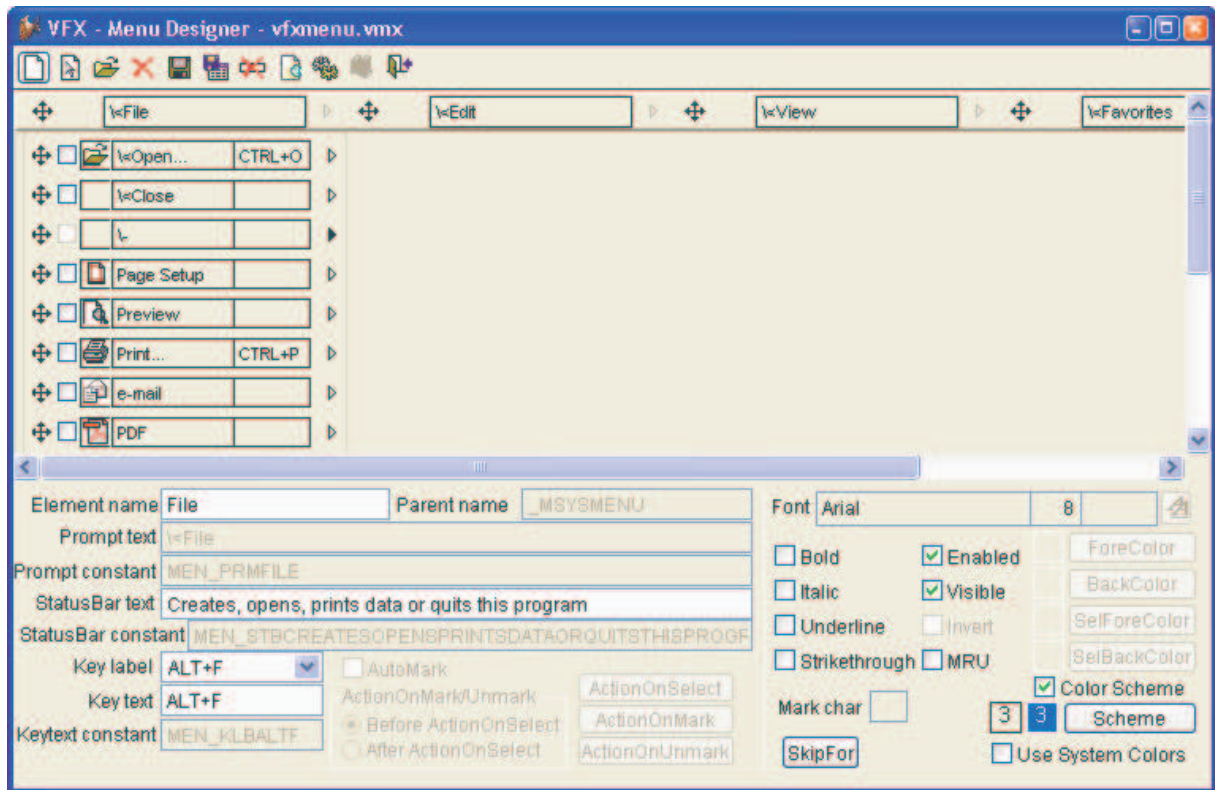
Click on finish to proceed.

Cancel < Back Next > Finish

Der Metadata Wizard erstellt die Tabelle „Datadict.dbf“. Dies ist eine freie Tabelle, in der die Struktur der SQL Server Datenbank inklusiv Constraints, benutzerdefinierten Datentypen, Regeln, Ansichten und gespeicherten Prozeduren gespeichert wird. Der Metadata Wizard durchsucht das aktive Projekt nach Verbindungen und analysiert die Struktur der Datenbank. Wenn die Tabelle „Datadict.dbf“ an die Kunden weitergegeben wird, wird die Struktur der dortigen Datenbank aktualisiert. Dabei wird wieder die bestehende Verbindung zum Zugriff auf die Datenbank verwendet.

Der VFX Menü-Designer

Der VFX Menü-Designer (VMD) ist ein Werkzeug zur schnellen Entwicklung von Menüs. Der VMD ist ein visueller Designer, in dem das Menü schon während der Entwicklung so angezeigt wird, wie es zur Laufzeit aussehen wird. Der VMD macht die Entwicklung einfacher und ermöglicht die schnelle Einstellung aller Menü-Eigenschaften im Gegensatz vom VFP Menü-Designer, der nicht alle Eigenschaften von Menüs unterstützt. Es können mehrsprachige Menüs erstellt werden, indem auf die entsprechende Schaltfläche in der Symbolleiste geklickt wird.



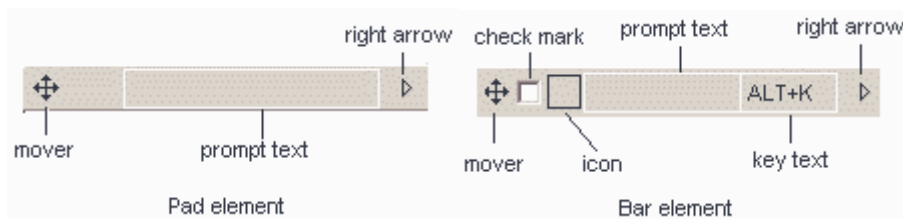
Ein in einem VFX-Projekt enthaltenes Menü kann direkt aus dem VFP-Projekt-Manager mit dem VMD geöffnet werden. Wahlweise können Menüs auch aus dem VMD heraus über das Öffnen-Symbol in der Symbolleiste oder über den entsprechenden Menüpunkt geöffnet werden. Im Öffnen-Dialog kann zwischen den Menütypen .mnx und .vmx gewechselt werden. Wenn ein Menü geöffnet wird, das noch nie mit dem VMD bearbeitet wurde, wird es automatisch in das .vmx-Format konvertiert.

Das geöffnete Menü kann visuell bearbeitet werden. Es können Einträge hinzugefügt und gelöscht werden und es können die Eigenschaften der einzelnen Einträge bearbeitet werden.

Neue Menü-Pads können durch einen Klick auf den Rechtspfeil, der sich rechts neben jedem Menüeintrag befindet, angelegt werden. Dadurch wird ein Untermenü angelegt. Einem Menü können neue Einträge hinzugefügt werden, indem auf den Pfeil nach unten unterhalb des Eintrags geklickt wird.

Ein Menüeintrag oder ein Menü-Pad können gelöscht werden, wenn sich der Fokus darauf befindet. Über den Menüpunkt löschen oder mit der Tastenkombination *Strg+Entf* wird der markierte Eintrag gelöscht.

Einige der Eigenschaften eines Menüeintrags können visuell eingestellt werden:



- Prompt text

Der angezeigte Text kann direkt eingetragen werden, wenn sich der Fokus auf dem jeweiligen Eintrag befindet. Die Textbox „*Prompt text*“ im unteren Teil des VMD dient nur zur Anzeige des aktiven Eintrags im visuellen Teil des Designers.

- *Key text*

Die Bezeichnung des Tastenschlüssels zeigt dem Anwender die Zugriffstaste oder Tastenkombination an, mit der der Eintrag ausgewählt werden kann. Die Bezeichnung sollte dem im unteren Teil des VMD gewählten Tastenschlüssels entsprechen.

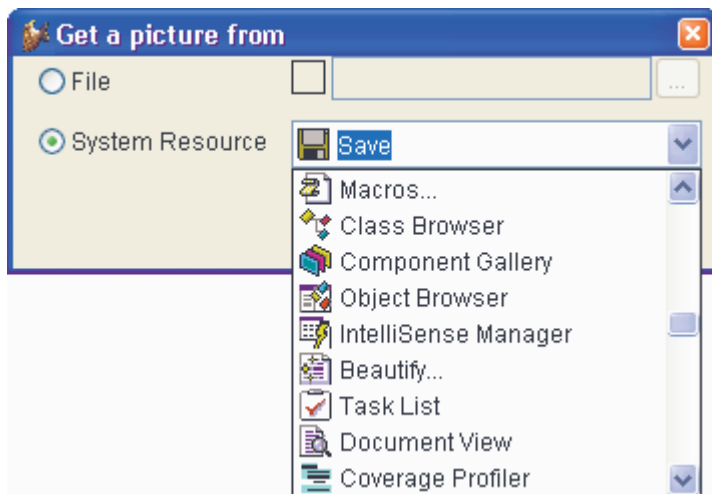
- *Check mark*

Damit sich ein Menüeintrag wie ein Kontrollkästchen verhält, muss bei „*AutoMark*“ eine Markierung gesetzt werden. Wenn zusätzlich eine Markierung bei „*Check mark*“ gesetzt wird, ist der Menüeintrag bereits beim Laden des Menüs markiert. Für Menüeinträge, die sich wie ein Kontrollkästchen verhalten, können zusätzliche Code-Teile ausgeführt werden, wenn das entsprechende Kontrollkästchen markiert wird („*ActionOnMark*“) bzw. wenn die Markierung aufgehoben wird („*ActionOnUnmark*“). Im Standard-VFP-Editorfenster kann der jeweilige Code bearbeitet werden.


Der Code, der bei *ActionOnMark* oder *ActionOnUnmark* eingegeben wird, kann wahlweise vor oder nach der *ActionOnSelect* ausgeführt werden. Um dieses Verhalten einzustellen, ist die entsprechende Option „*Before ActionOnSelect*“ oder „*After ActionOnSelect*“ auszuwählen.

- *Icon*

Jedem Eintrag in einem Menü kann ein Symbol zugeordnet werden. Dieses Symbol kann aus den in VFP integrierten Systemressourcen ausgewählt werden oder es kann eine Datei verwendet werden. Durch einen Klick auf das schwarzumrandete Kästchen kann ein Symbol mithilfe des „*Get a picture from*“-Dialogs ausgewählt werden. In diesem Dialog kann zwischen einer Datei und einem Symbol aus den VFP Systemressourcen gewählt werden.



Wenn einem Menüeintrag ein Symbol zugeordnet ist und sich dieser Menüeintrag wie ein Kontrollkästchen verhalten soll, dient das Symbol als Markierung. Wenn der Eintrag markiert wird, erscheint das Symbol eingedrückt. Wenn die Markierung aufgehoben wird, erscheint das Symbol normal.

Die Position der einzelnen Einträge innerhalb des Menüs kann per Drag & Drop verändert werden. Für diesen Vorgang ist der Vierwegepfeil , der sich links neben allen Einträgen befindet festzuhalten. In einigen Fällen sind Verschiebeoperationen nicht möglich. Ein Menü-Pad kann nicht in einen Menüeintrag umgewandelt werden und umgekehrt. Außerdem ist es nicht möglich einen Menüeintrag in ein Untermenü zu verschieben.

Weitere Eigenschaften der Menüeinträge können im unteren Teil des Menü-Designers eingestellt werden. Dazu gehören der Zeichensatz, die Vordergrund- und Hintergrundfarbe, eine Meldung, die in der VFP-Statusbar angezeigt wird sowie der Name einer Konstanten, die verwendet wird, wenn ein mehrsprachiges Menü erstellt wird. Alle Änderungen werden unmittelbar im aktiven Element angezeigt.

Mit der Schaltfläche „*ActionOnSelect*“ kann in einem Editor-Fenster die auszuführende Aktion eingegeben werden. Über die Schaltfläche „*SkipFor*“ kann eine Bedingung eingegeben werden. Wenn diese Bedingung .T. liefert, kann der dazugehörige Menüeintrag nicht ausgewählt werden.

Die eingestellten Eigenschaften beziehen sich immer auf den aktiven Menüeintrag. Neue Menüeinträge erben die Eigenschaften des zuvor ausgewählten Eintrags.

Der Zeichensatz kann über die Schaltfläche „Font“ ausgewählt werden. Der Standard-Windows-Dialog zur Auswahl eines Zeichensatzes erscheint. In diesem Dialog können insbesondere die Schriftart und die Schriftgröße sowie der Schriftschnitt ausgewählt werden.

Zu jeder Zeit kann eine Vorschau des Menüs angezeigt werden, indem in der Symbolleiste oder im VMD-Menü „Preview“ gewählt wird.

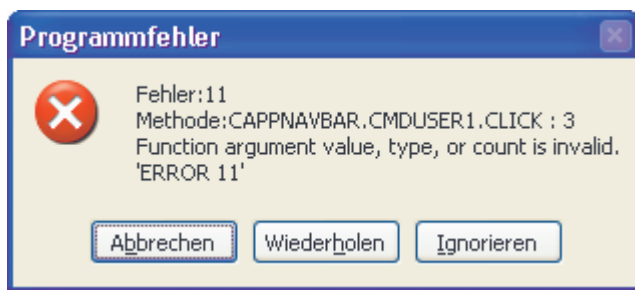
Eigenschaften der Klasse cApplication

Die Klasse des Applikationsobjekts wurde um eine Reihe neuer Eigenschaften zur Steuerung des Verhaltens der Applikation im Fehlerfall, zur Verwendung der Produktaktivierung und zur Installation eines Postscript-Druckertreibers, der zur Erstellung von PDF-Dateien benötigt wird, erweitert. Die Werte dieser Eigenschaften können in Vfxmain.prg unter *DEFINE CLASS capplication AS cfoxpath* eingestellt werden.

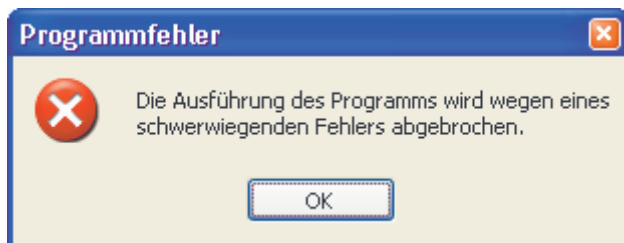
nAppOnErrorBehavior – Diese Eigenschaft steuert das Verhalten der Applikation im Fehlerfall.

0 – Laufzeitfehler werden ignoriert.

1 – Es wird eine Fehlermeldung angezeigt (Standardwert).



2 – Die Ausführung der Applikation wird nach Anzeige eines Hinweises beendet.



ErrorDetailLevel – Diese Eigenschaft steuert welche Informationen im Fehlerfall in der Tabelle Vfxlog.dbf protokolliert werden.

0 – Nur die Fehlermeldung aber keine Information über den Aufrufstapel.

1 – Die Fehlermeldung und Informationen über den Aufrufstapel (Standardwert).

2 – Vollständige, detaillierte Fehlerinformationen.

PSPrinterToInstall – Diese Eigenschaft enthält den Namen des Standard-Postscript-Druckertreibers. Dieser Druckertreiber wird automatisch installiert, wenn noch kein Postscript-Druckertreiber installiert ist und die Applikation einen Postscript-Druckertreiber braucht um eine PDF-Datei zu erstellen. Der Standardwert ist "HP DeskJet 1200C/PS".

cConnectionCheckURL – Diese Eigenschaft enthält die Adresse einer Internetseite, die verwendet wird um zu testen, ob eine Internet-Verbindung besteht. Diese Eigenschaft wird benötigt wenn Ghostscript nicht installiert ist. Ghostscript wird bei Bedarf automatisch aus dem Internet heruntergeladen und installiert. Ghostscript wird verwendet um Postscript-Dateien in PDF-Dateien umzuwandeln. Wenn keine Verbindung mit dem Internet besteht und auch keine DFÜ-Netzwerkverbindung eingerichtet ist, wird von VFX ein Eintrag im DFÜ-Netzwerk angelegt. Alle Eigenschaften der DFÜ-Verbindung können vom Entwickler vorgegeben werden. Der Anwender kann bei Bedarf in einem Dialog die Telefonnummer, den Benutzernamen und das Kennwort ändern.

UseActivation – Über diese Eigenschaft wird die Produktaktivierung ein- bzw. ausgeschaltet. Diese Eigenschaft kann im VFX - Application Wizard eingestellt werden, wenn ein neues Projekt erstellt wird. Später kann der Eintrag in Vfxmain.prg geändert werden. Der Standardwert ist .F., die Produktaktivierung wird nicht verwendet.

ActivationType – Wenn diese Eigenschaft auf .T. gesetzt wird, überprüft die Klasse cVFXActivate ob die Datei „FirstInstall.txt“ existiert, wenn die Applikation gestartet wird. Diese Eigenschaft kann im VFX Application Wizard eingestellt werden, wenn ein neues Projekt erstellt wird. Später kann der Eintrag in Vfxmain.prg geändert werden. Der Standardwert ist .F., es wird nicht auf das Vorhandensein der Datei „FirstInstall.txt“ geprüft.

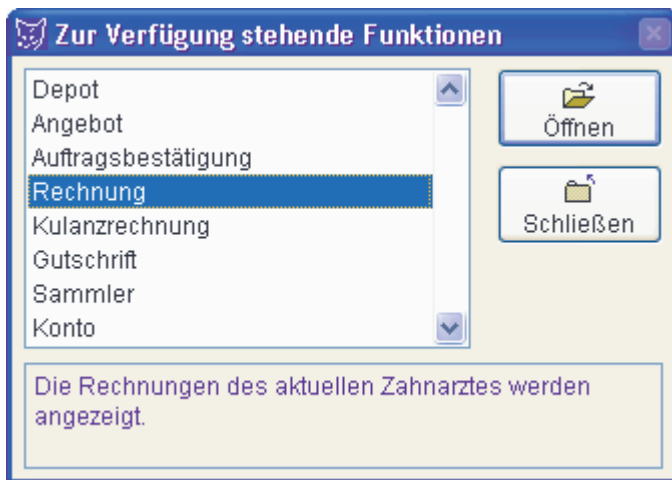
Wichtige VFX-Methoden

Valid

VFX bietet eine Valid-Methode auf Formularebene. Diese Methode wird immer aufgerufen, wenn die Daten des Formulars gespeichert werden sollen. Hier sollten also alle Validierungen untergebracht werden. Wenn aus dieser Methode der Wert .F. zurückgegeben wird, wird der Speichervorgang nicht fortgesetzt und das Formular bleibt im Bearbeitungsmodus. Durch Rückgabe von .T. werden die Daten gespeichert.

Onmore

Mithilfe dieser Methode ist es insbesondere möglich Child-Formulare aufzurufen. Ein fertiger Template-Code kann auf Wunsch vom VFX – Form Builder im Formular eingetragen werden. Je nach Anwendungsfall brauchen nur noch wenige Werte dieser Methode vom Entwickler angepasst werden.



Über die Onmore-Methode wird zur Laufzeit ein Dialog angezeigt, in dem der Benutzer das aufzurufende Child-Formular auswählen kann.

Onpostinsert

Diese Methode wird unmittelbar nach dem Anfügen eines neuen Datensatzes aufgerufen, noch bevor der Benutzer die Möglichkeit zur Bearbeitung der Daten erhält.

Hier können also Standardvorgaben in den Feldern eingetragen werden. Diese Methode bietet sich auch an, um Primärschlüssel zu vergeben.

Onrecordmove

Jedes Mal, wenn der Satzzeiger bewegt wird, wird diese Methode aufgerufen. Hier können Werte angezeigt oder aktualisiert werden, die nicht aus der Datenbank stammen.

Einstellungen zur Datenanbindung des TreeView-Steuerlements

IDFieldName – Hier wird der Name des Feldes mit dem Primärschlüssel der Bearbeitungstabelle eingetragen.

ParentIDFieldName – Diese Eigenschaft enthält den Namen des Feldes, in dem der Primärschlüssel des Parent-Datensatzes gespeichert ist.

NodeText – Hier kann entweder der Name eines Feldes, das einen Beschreibungstext enthält eingetragen werden oder es wird ein Ausdruck eingetragen, der zur Laufzeit evaluiert wird und dessen Rückgabewert als Bezeichnung in der Baumstruktur angezeigt wird. Wenn ein Feldname verwendet wird, kann dem Anwender erlaubt werden die Bezeichnung direkt im Treeview-Steuerlement zu ändern. Dies hängt vom Wert der Eigenschaft *AllowNodeRename* ab. Wenn *AllowNodeRename* auf .T. gesetzt ist, kann der Anwender die Bezeichnungen im Treeview-Steuerlement ändern. Dabei werden die Daten im zugrunde liegenden Tabellenfeld automatisch aktualisiert.

AllowNodeRename – Über diese Eigenschaft wird gesteuert, ob der Anwender die Bezeichnung im Treeview-Steuerlement ändern kann. Die Bearbeitung der Bezeichnung im Treeview-Steuerlement ist nur möglich, wenn die Bezeichnung auf einem einzelnen Tabellenfeld basiert. Dieses Tabellenfeld wird bei der Bearbeitung automatisch aktualisiert.

Layout-Einstellungen des TreeView-Steuerlements

Diese Einstellungen entsprechen denen des TreeView-ActiveX-Steuerlements.

Style: 0 - tvwStyleText
1 - tvwStylePictureText
2 - tvwStylePlusMinusText
3 - tvwStylePlusMinusPictureText
4 - tvwStyleLinesText
5 - tvwStyleLinesPictureText
6 - tvwStyleLinesPlusMinusText
7 - tvwStyleLinesPlusMinusPictureText

Appearance: 0 - ccFlat
1 - cc3D

BorderStyle: 0 - ccNone
1 - ccFixedSingle

Indentation: Diese Eigenschaft bestimmt die Breite des Einzugs der Knoten.

Die Klasse cDownload

Diese Klasse ermöglicht das Herunterladen von Dateien aus dem Internet. Bei Bedarf können die heruntergeladenen Dateien ausgeführt werden und es können weitere Aktionen ausgeführt werden. Insbesondere ist hierdurch die Installation von Programmen aus dem Internet möglich.

Durch die Verwendung von Makros und die Execmacro-Funktion von VFP kann diese Klasse sehr vielseitig eingesetzt werden.

Makros sind Zeichenketten, die eine Folge von Befehlen aus der Makrosprache enthalten. Eigene Makros können erstellt werden. Ein Beispiel ist in der Tabelle Vfxsys.dbf im Feld *Install_GS* zu finden. Mit diesem Makro wird das Programm Ghostscript aus dem Internet heruntergeladen und installiert.

Diese Klasse verwendet die in der Eigenschaft *goProgram.cConnectionCheckURL* gespeicherte Internetseite um zu überprüfen, ob eine Internetverbindung besteht. Bei Bedarf wird eine Verbindung automatisch hergestellt. Wenn im DFÜ-Netzwerk keine Verbindung eingetragen ist, wird ein neuer Eintrag hinzugefügt. Die Verbindungsinformationen kann der Entwickler in den Eigenschaften vorgeben. Der Anwender kann die Telefonnummer, den Benutzernamen und das Kennwort für die neue Verbindung bei Bedarf in einem Dialog ändern.

Eigenschaften

LastErrorNo – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

LastErrorTest – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

Methoden

ExecMacro (vcMacro, InNoRun)

vcMacro – Skript der Makrosprache, das ausgeführt werden soll.

InNoRun – Wenn diese Eigenschaft auf .T. gesetzt wird, wird die heruntergeladene Datei nicht ausgeführt.

Befehle der Makrosprache

„D:“ *URL*

Unter dieser Internetadresse ist die herunterzuladende Datei zu finden. Dieser Befehl führt die Datei nach dem erfolgreichen Herunterladen aus, wenn die Eigenschaft *InNoRun* auf .F. gesetzt ist.

„C:“ *nTimeout; lPartial; lTopLevelForm; lResultOnError; SearchedString*

Wartet bis das Fenster mit dem Titel *SearchedString* erscheint.

nTimeout – Timeout in Sekunden. Wenn das erwartete Formular nicht innerhalb dieser Zeitspanne erscheint, wird ein Timeout-Fehler erzeugt.

lPartial – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, reicht es, wenn der übergebene Titel einem Teil des Fensternamens entspricht. Wenn diese Eigenschaft auf .F. gesetzt ist, muss der übergebene Titel exakt dem Namen des Fensters entsprechen.

lTopLevelForm – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, wird der Fenstername nur in Top-Level-Fenstern gesucht.

lResultOnError – Mit dieser Eigenschaft wird das Verhalten des Skripts gesteuert, falls das Fenster nicht innerhalb der vorgegebenen Zeitspanne gefunden wurde. Wenn das Fenster für die weitere Ausführung des Skripts zwingend erforderlich ist, muss nach Ablauf der vorgegebenen Zeitspanne die Ausführung des Skripts abgebrochen werden. In diesem Fall muss der Wert von *lResultOnError* auf .F. gesetzt werden. Wenn die Ausführung des Skripts unabhängig vom Vorhandensein des Fensters nach der vorgegebenen Zeitspanne fortgesetzt werden soll, muss *lResultOnError* auf .T. gesetzt werden.

SearchedString – Bezeichnung, die in einem Fensternamen gesucht wird.

„W:“ *nTimeout; lPartial; lTopLevelForm; lResultByError; SearchedString*

Es wird gewartet bis das Fenster, das die angegebene Zeichenkette im Titel enthält, geschlossen ist.

nTimeout – Timeout in Sekunden. Wenn das erwartete Fenster innerhalb dieser Zeitspanne nicht geschlossen ist, wird ein Timeout-Fehler ausgelöst.

lPartial – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, reicht es, wenn der übergebene Titel einem Teil des Fensternamens entspricht. Wenn diese Eigenschaft auf .F. gesetzt ist, muss der übergebene Titel exakt dem Namen des Fensters entsprechen.

lTopLevelForm – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, wird der Fenstername nur in Top-Level-Fenstern gesucht.

lResultOnError – Mit dieser Eigenschaft wird das Verhalten des Skripts gesteuert, falls das Fenster nicht innerhalb der vorgegebenen Zeitspanne gefunden wurde. Wenn das Fenster für die weitere Ausführung des Skripts zwingend erforderlich ist, muss nach Ablauf der vorgegebenen Zeitspanne die Ausführung des Skripts abgebrochen werden. In diesem Fall muss der Wert von *lResultOnError*

auf .F. gesetzt werden. Wenn die Ausführung des Skripts unabhängig vom Vorhandensein des Fensters nach der vorgegebenen Zeitspanne fortgesetzt werden soll, muss *lResultOnError* auf .T. gesetzt werden.

SearchedString – Eine Zeichenkette nach der im Titel eines Fensters gesucht wird.

„X:“

Schließt das Top-Level-Fenster. Mit dem „C:“-Befehl muss zuvor sichergestellt werden, dass das gewünschte Fenster sichtbar ist.

„K:“ *nKeyCode1; nKeyCode2; ...*

Die aufgeführten Tastenschlüssel werden in den Windows-Tastaturpuffer übertragen.

„U:“ *URL*

Von dieser Internetadresse wird das Herunterladen ausgeführt. Die heruntergeladene Datei wird unabhängig vom Wert der Eigenschaft *InNoRun* nicht ausgeführt.

Beispiel

Beschreibung der Installation von Ghostscript:

D: <ftp://mirror.cs.wisc.edu/pub/mirrors/ghost/AFPL/gs811/gs811w32.exe>

Lädt die Datei *gs811w32.exe* aus dem Internet herunter und führt sie anschließend aus.

C: 30; .F.; .F.; .F.; WinZip Self-Extractor - gs811w32.exe

Wartet bis das Fenster mit dem Titel „WinZip Self-Extractor - gs811w32.exe“ erscheint.

K: 43

Sendet den Tastenschlüssel „Eingabetaste“ an das aktive Fenster. Dadurch wird das Entpacken der Dateien ausgelöst.

C: 60; .F.; .F.; .F.; AFPL Ghostscript Setup

Wartet bis das Fenster mit dem Titel „AFPL Ghostscript Setup“ erscheint.

K: 43

Sendet den Tastenschlüssel „Eingabetaste“ an das aktive Fenster. Dadurch wird die Installation von Ghostscript gestartet.

W: 240; .F.; .F.; .F.; AFPL Ghostscript Setup Log

Wartet solange das Fenster „AFPL Ghostscript Setup Log“ geöffnet ist. Dieses Fenster zeigt den Fortschritt der Installation an und die Skriptausführung muss warten, bis dieser Vorgang beendet ist.

C: 30; .T.; .T.; .T.; Ghostscript

Wartet bis das Fenster mit dem Titel „Ghostscript“ erscheint. Dieses Fenster zeigt die Nachricht an, dass die Installation erfolgreich war.

X:

Schließt das letzte Fenster.

Hiermit ist die Installation von Ghostscript beendet.

Die Klasse cCreatePDF

Diese Klasse erstellt Berichtsausgaben im PDF-Format. Als Parameter werden der Aliasname des zu verwendenden Cursors, der Name der zu erstellenden PDF-Datei, der Name der Berichtsdatei sowie eine optionale For-Klausel übergeben.

Um eine PDF-Datei erstellen zu können, müssen Ghostscript und ein Postscript-Druckertreiber auf dem jeweiligen Computer installiert sein. Diese Klasse prüft, ob Ghostscript bereits installiert ist. Sollte dies nicht der

Fall sein, wird Ghostscript automatisch aus dem Internet heruntergeladen und installiert. Für das Herunterladen aus dem Internet wird die Klasse `cDownload` verwendet. In dem Memofeld *Install_gs* aus der Tabelle *Vfxsys.dbf* befindet sich das Skript, das zum Herunterladen und zur Installation von Ghostscript verwendet wird. In der Beschreibung der Klasse `cDownload` befinden sich weitere Hinweise.

Wenn kein Postscript-Druckertreiber installiert ist, installiert diese Klasse automatisch den Druckertreiber, dessen Name in der Eigenschaft *goProgram.PSPrinterToInstall* hinterlegt ist. In der Regel sind hierfür keine Benutzereingaben erforderlich.

Der Bericht wird über den Postscript-Druckertreiber ausgegeben und in einer Datei gespeichert. Das Programm Ghostscript wandelt diese Postscript-Datei in eine PDF-Datei um.

Eigenschaften

LastErrorNo – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

LastErrorText – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

Methoden

AddAttachment (*tsAlias*, *tcFileName*, *tcReport*, *tcFor*)

Fügt dem `CreatePDF`-Objekt Informationen über eine Datei hinzu. Es wird automatisch eine PDF-Datei zum dem als Parameter übergebenen Bericht erstellt. Ein weiterer Ausdruck kann als Parameter angegeben werden. Dieser Ausdruck dient dazu die Daten des Berichts zu filtern.

Create_PDF(*tcAlias*, *tcRezFile*, *tcFRXName*, *tcFor*)

tcAlias – Aliasname, der für die Berichtsausgabe verwendet wird.

tcRezFile – Vollständiger Pfadname der zu erstellenden PDF-Datei.

tcFRXName – Name der Berichtsdatei, die zur Erstellung der PDF-Datei verwendet wird.

tcFor – For-Klausel zur Filterung der zu exportierenden Daten.

Diese Methode gibt den Wert `.T.` zurück, wenn die PDF-Datei erfolgreich erstellt werden konnte. `.F.` wird zurückgegeben, wenn die PDF-Datei nicht erstellt werden konnte. In diesem Fall sind die Nummer und die Beschreibung des aufgetretenen Fehlers in den Eigenschaften *LastErrorNo* und *LastErrorText* gespeichert.

Die Klasse `cEmail`

Diese Klasse gibt dem Entwickler die Möglichkeit E-Mails zu versenden. Es müssen nur wenige Parameter der Methode *Send_Email_Report* übergeben werden um eine Berichtsausgabe im PDF-Format als E-Mail-Anhang versenden zu können.

Eigenschaften

LastErrorNo – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

LastErrorText – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

oEmail_Attachment – Diese Eigenschaft wird nur intern verwendet. Sie enthält eine Collection der Anhänge.

Methoden

AddAttachment (*tsAlias*, *tcFileName*, *tcReport*, *tcFor*)

Fügt dem E-Mail-Objekt Informationen über einen E-Mail-Anhang hinzu, der mit der nächsten E-Mail gesendet wird. Die Informationen über alle vorzubereitenden PDF-Anhänge werden in der Eigenschaft *oEmail_Attachment* gespeichert. Wenn der Aliasname einer geöffneten Tabelle oder Ansicht angegeben und der Name einer Berichtsdatei übergeben wird, wird diese Klasse automatisch eine PDF-Datei zu dem Bericht erstellen. Es kann ein weiterer Ausdruck als Parameter angegeben werden, der dazu verwendet wird die Daten des Berichts zu filtern. Wenn kein Aliasname angegeben wird und keine Tabelle im aktuellen Arbeitsbereich geöffnet ist, nimmt die Klasse an, dass ein Dateianhang vorbereitet wurde. In diesem Fall muss die Datei existieren, wenn die Methode *Send_Email_Report* aufgerufen wird.

tcAlias – Aliasname, der für die Berichtsausgabe und für den PDF-Export verwendet wird.

tcRezFile – Name des Dateianhangs (wenn eine PDF-Datei erstellt wird, wird dies der Name der PDF-Datei).

tcFRXName – Name der Berichtsdatei, aus der die PDF-Datei erstellt wird.

tcFor – For-Klausel mit der die Berichtsdaten für die PDF-Ausgabe gefiltert werden.

Send_Email_Report (*tcEmail*, *tcSubject*, *tcText*)

Sendet eine E-Mail. Wenn die E-Mail mit Anhängen versendet werden soll, müssen diese vorher mit der Methode *AddAttachment* angefügt werden.

tcEmail – Adresse des E-Mail-Empfängers.

tcSubject – Betreff der E-Mail.

tcText – Text der E-Mail.

ClearAttachment

Löscht alle E-Mail-Anhänge.

Die Methode *AddAttachment* kann entsprechend der Anzahl der benötigten Anhänge beliebig oft aufgerufen werden. Es werden die Aliasnamen der Tabellen oder Ansichten, die Namen der zu erstellenden Dateien, die Namen der Berichtsdateien und eventuell zu verwendende For-Klauseln als Parameter übergeben. Dann wird die Methode *Send_Email_Reports* aufgerufen. Alle PDF-Dateien werden erstellt und als E-Mail-Anhänge versendet. Auch die Dateien, die zuvor vorbereitet wurden und als Anhang versendet werden sollen, werden an die E-Mail angehängt.

Die Klasse cArchive

Diese Klasse dient der Datensicherung und Datenwiederherstellung. Die Daten werden in Zip-Archiven gesichert. Der Name des Archivs wird aus dem Namen des Datenordners und dem aktuellen Datum in ANSI-Form zusammengesetzt. Wenn zum Beispiel der Datenordner „Data“ heißt und die Datensicherung am 4. November 2003 durchgeführt wird, heißt das Archiv Data20031104.zip.

Eigenschaften

OverrideFile – Mit dieser Eigenschaft wird festgelegt was passiert, wenn eine Datei mit dem gleichen Namen schon vorhanden ist.

0 – Vorgang abbrechen, wenn bereits eine Datei mit dem gleichen Namen existiert.

1 – Wenn eine Datensicherung durchgeführt wird, werden neue Dateien dem Archiv hinzugefügt und bestehende Dateien werden aktualisiert. Wenn eine Wiederherstellung durchgeführt wird, werden existierende Dateien nicht überschrieben.

2 – Wenn eine Datensicherung durchgeführt wird, wird ein bestehendes Archiv überschrieben. Wenn eine Wiederherstellung durchgeführt wird, werden existierende Dateien überschrieben.

OperationSuccessfully – Enthält das Ergebnis der letzten Aktion.

.T. – wenn die Aktion erfolgreich ausgeführt werden konnte.

.F. – wenn die Aktion nicht ausgeführt werden konnte.

Methoden

CreateArchive (*lcFileLocation*, *lcMask*, *lcArchFilePathName*)

lcFileLocation – Vollständiger Pfad zu dem Ordner, dessen Inhalt gesichert werden soll.

lcMask – Zu sichernde Dateien, Beispiel: „*.DBF;*.FPT;*.CDX“.

lcArchFilePathName – Vollständiger Pfadname der zu erstellenden Archivdatei.

Rückgabewert: .T. – wenn die Aktion erfolgreich ausgeführt werden konnte, .F. – wenn die Aktion nicht ausgeführt werden konnte.

ZipProgress (*tcCurrentOperatedFile*, *nState*, *nAllFilesSize*, *nZIPedFilesSize*, *nArchiveCurrentSize*) Callback-Funktion der *CreateZipArchive*-Funktion (in VFX.fl).

tcCurrentOperatedFile – Der Name der Datei, die dem Archiv hinzugefügt wird.

nState – Aktuelle Aktion:

- 1 – Datei existiert
- 2 – Datei wird dem Archiv hinzugefügt
- 3 – Datei erfolgreich dem Archiv hinzugefügt
- 4 – Datei konnte dem Archiv nicht hinzugefügt werden
- 5 – Archivierungsvorgang erfolgreich beendet
- 6 – Archivierungsvorgang nicht erfolgreich beendet
- 7 – Keine Dateien zu archivieren

nAllFilesSize – Die Größe aller zu archivierenden Dateien.

nZIPedFilesSize – Die Größe der dem Archiv bereits hinzugefügten Dateien.

nArchiveCurrentSize – Die aktuelle Größe des Archivs.

Rückgabewert: 0 – Abbruch der Aktion. 1 – Fortsetzen Dateien dem Archiv hinzuzufügen und existierende Dateien zu überschreiben. 2 – Bestehende Archivdatei überschreiben

EextractFromArchive(*lcArchFileForExtract*, *lcPathForExtract*)

lcArchFileForExtract – Vollständiger Pfadname der zu entpackenden Zip-Datei.

lcPathForExtract – Zielordner, in den die Dateien entpackt werden sollen.

UnZipProgress (*tcCurrentOperatedFile*, *nState*, *nArchiveFilesSize*, *nUnZIPedFilesSize*)

Callback-Funktion der *ExtractZipArchive*-Funktion (in VFX.fl).

tcCurrentOperatedFile – Name der aktuell entpackten Datei aus dem Archiv.

nState – Aktuelle Aktion

- 1 – Datei existiert bereits
- 2 – Datei wird entpackt
- 3 – Datei entpacken beendet
- 4 – Datei konnte nicht entpackt werden
- 5 – Entpacken des Archiv erfolgreich abgeschlossen
- 6 – Entpacken des Archiv nicht erfolgreich abgeschlossen

nArchiveFilesSize – Größe des Archivs

nUnZIPedFilesSize – Größe des Teils des Archivs, das bereits entpackt wurde.

Rückgabewert: 0 – Abbruch der Aktion. 1 – Aktuelle Datei nicht entpacken. 2 – Vorhandene Datei überschreiben.

Die Klasse cPickAlternate

Ähnlich zum cPickField-Steurelement kann die Klasse cPickAlternate verwendet werden um eine Benutzereingabe zu verifizieren. Es kann eine Auswahlliste aufgerufen werden, die dem Anwender erlaubt einen Wert aus einer Liste auszuwählen. Bei Verwendung der Klasse cPickAlternate wird der Primärschlüssel des ausgewählten Datensatzes in der Bearbeitungstabelle gespeichert während der Benutzer einen Wert aus einem anderen Feld aus der Auswahltabelle angezeigt bekommt.

Das cPickAlternate-Steurelement ist einer Combobox zu bevorzugen, wenn aus einer Tabelle mit vielen Datensätzen ausgewählt werden soll. Der Einsatz ist auch sinnvoll, wenn der vom Anwender eingegebene Wert nicht dem Schlüssel der Auswahltabelle entspricht. Das Ziel dieser Klasse ist es dem Anwender eine einfach zu bedienende Schnittstelle zu geben, die es erlaubt ihm bekannte Werte einzugeben anstelle von vom Programm generierten Primärschlüsseln. Der vom Anwender eingegebene Wert wird verwendet um den dazugehörigen Datensatz in der Auswahltabelle zu finden. Wenn der gesuchte Datensatz gefunden ist, wird als Rückgabewert der Primärschlüssel an das cPickAlternate-Steurelement zurückgegeben.

Diese Klasse basiert auf der Klasse cPickField und erbt alle ihre Eigenschaften und Methoden. Zusätzlich hat diese Klasse die neue Eigenschaft *cControlSourceInternalKey* in die der Name des Feldes der Bearbeitungstabelle mit dem Fremdschlüssel eingetragen wird. Dieser Fremdschlüssel entspricht dem Primärschlüssel aus der Auswahltabelle.

Mithilfe des VFX – CPickAlternate Builder können die Eigenschaften dieser Klasse einfach eingestellt werden.

Pick Table Name – Hier kann der Name der Auswahltabelle aus einer der Datenquellen der Datenumgebung ausgewählt werden.

Pick Table Index Tag – Dies ist der Name des Indexschlüssels, der verwendet wird um in der Auswahltabelle zu suchen. Dieser Indexschlüssel entspricht dem Wert des Eingabefeldes.

CPickAlternate.txtField.ControlSource – Die Controlsource des Eingabefeldes. Dieses Feld muss aus der Auswahltabelle stammen.

CpickAlternate.txtDesc.ControlSource – Der Name des Beschreibungsfeldes. Der Wert wird nach der erfolgreichen Überprüfung der Benutzereingabe im Beschreibungsfeld angezeigt. Dieses Feld stammt ebenfalls aus der Auswahltabelle.

Return Field Name (Code) – Der Name des Feldes mit dem vom Anwender eingegebenen Wert aus der Auswahltable. In der Regel entspricht dieser Feldname dem Namen, der in *txtField.ControlSource* angegeben ist. Hier ist nur der Feldname ohne den Tabellennamen anzugeben. Der Wert dieses Feldes muss vom Typ „Zeichen“ sein. Gegebenenfalls ist der Wert mit *TRANSFORM()* in einen Zeichentyp umzuwandeln.

Return Field Name (Description) – Der Name des Feldes mit der Beschreibung, die aus der Auswahltable zurückgegeben wird. Es kann auch ein Ausdruck zurückgegeben werden. Der Wert wird im Beschreibungsfeld angezeigt. Der Wert muss vom Typ „Zeichen“ sein. Gegebenenfalls ist der Wert mit *TRANSFORM()* in einen Zeichentyp umzuwandeln.

Return Field Name (Internal Key) – Der Name des Feldes aus der Auswahltable, das den Primärschlüssel enthält. Über dieses Feld wird die Beziehung von der Bearbeitungstabelle zur Auswahltable in der Datenumgebung hergestellt.

Control Source Internal Key – Der Name des Feldes aus der Bearbeitungstabelle, das den Primärschlüssel enthält. Dieses Feld enthält den Fremdschlüssel aus der Auswahltable.

Portierung einer Anwendung von VFX 7 nach VFX 8.0

Bestehende VFX 7-Anwendungen können einfach nach VFX 8.0 portiert werden. Da sich zwischen VFX 7 und VFX 8.0 vieles geändert hat, ist es empfehlenswert zunächst ein neues, leeres Projekt mit VFX 8.0 zu erstellen. Dieses neue Projekt wird in einem neuen Verzeichnis erstellt, bekommt aber den Namen und insbesondere auch den Datenbanknamen des alten Projekts. Alle Eigenschaften im VFX – Application Wizard werden wie beim alten VFX 7 Projekt eingetragen.

Im neuen VFX 8.0-Projekt stehen damit automatisch die neuen Menüs von VFX 8.0, die erweiterten Include-Dateien sowie die erweiterten Strukturen der freien Tabellen zur Verfügung. Wenn Sie das Menü *Vfxmenu* Ihres alten Projektes erweitert hatten, machen Sie die Erweiterungen in dem neuen Menü bitte manuell. Nur so bleiben die erweiterten Menüeinträge von VFX 8.0 erhalten.

Prüfen Sie das neue *VFXMAIN.PRG* und machen Sie von Hand die für Ihr Projekt erforderlichen Änderungen. Folgen Sie der Dokumentation im neuen *VFXMAIN.PRG*, um die Vorlage an Ihre spezifischen Bedürfnisse anzupassen.

Aus dem VFX 7-Projekt kann jetzt die Datenbank in den Data-Ordner des neuen Projekts kopiert werden. Die freien VFX-Tabellen werden im VFX 8.0-Projekt mit *USE* geöffnet und mit *APPEND FROM* werden die Daten aus dem alten VFX 7-Projekt der jeweiligen Tabelle angefügt. Auf diese Weise müssen die Daten der Tabellen *Vfxfopen.dbf*, *Vfxsys.dbf*, *Vfxsysid.dbf* und *Vfxusr.dbf* geholt werden.

Alle Formulare und Berichte werden aus den jeweiligen Ordnern, Form bzw. Report, in das neue Projekt kopiert und manuell dem VFX Projekt-Manager hinzugefügt. Schließlich wird noch die Datei *Applfunc.prg* aus dem *Program*-Ordner und die Dateien *Appl.vc** aus dem *Lib*-Ordner in das neue Projekt kopiert.

Wenn Sie eigene Konstanten in Include-Dateien abgelegt haben, kopieren Sie Ihre Include-Dateien in den *Include*-Ordner des neuen Projektes. Überschreiben Sie nicht die Include-Dateien *Vfxmsg.h* und *Vfxtxt.h*, da diese zahlreiche neue Konstanten enthalten, die von VFX 8.0 benötigt werden.

Nach dem Kompilieren aller Dateien ist die Anwendung mit VFX 8.0 einsatzbereit.

Schlusswort

Wie wir gesehen haben, stellt VFX 8.0 eine vollständige Entwicklungsumgebung bereit, die keine Wünsche offen lässt. Alle wesentlichen Einstellungen an VFX-Klassen, insbesondere an den Formulklassen, können mit reentranten Generatoren durchgeführt werden.

Da VFX mit Quellen geliefert wird und selbst mit VFP programmiert ist hat der Entwickler unbegrenzte Freiheit eigene Erweiterungen oder Anpassungen an eigene Bedürfnisse vorzunehmen.

Die mit VFX erstellten Applikationen vermitteln dem Anwender einen sehr professionellen Eindruck und eine weitgehend Office-kompatible Bedienung.

VFX bietet mit all dem ein unschlagbares Preis-/Leistungsverhältnis. Es bietet jedem Programmierer eine Fundgrube an Ideen und eine Vielzahl von fertigen Problemlösungen.