Visual.NET Extensions – Tutorial

“vdxPickField – Fremdschlüsselvalidierung mit Auswahlliste“

Das umfangreiche Applikationsentwicklungs-Framework für die einfache Entwicklung von Microsoft Visual Studio.NET Datenbank-Applikationen!

Devigus Engineering AG
Grundstrasse 3
CH-6343 Rotkreuz
Internet: http://www.devigus.com
Email: deag@devigus.com

Version: 2.0
Letztes Update: 17.11.2003

Inhaltsverzeichnis

1 *vdxPickField*-Szenario am Beispiel der Personenverwaltung

Dieses Tutorial beschreibt den Einsatz des *vdxPickField*-Controls. Das PicklField Control erlaubt Fremdschlüsselvalidierung mit zusätzlicher Auswahl-Funktionalität. Analog zu den übrigen Tutorials wird anhand der Beispiel-Applikation ein konkreter Anwendungsfall implementiert, welcher die Verwendung der *vdxPickField*-Klasse exemplarisch beschreibt.

Damit eine Person einer Adresse zugeordnet werden kann, muss ein entsprechendes Control in die Klasse *DataDetailPerson* (siehe *vdxExplorerTree - Erweiterte Datenmanipulation* - Tutorial) eingefügt werden. Die aktuelle Person wird dann der ausgewählten Adresse zugewiesen und entsprechend in der Adressenverwaltung unter der Adresse als Person aufgeführt. Die Abbildung 1 zeigt das in diesem Tutorial zu implementierende *vdxPickField*-Control. Dieses befindet sich an oberster Stelle (rote Markierung) und mit einem Klick auf den *Browse*-Button (…) kann die dazugehörige Auswahl (Picklist) Funktionalität aufgerufen werden:

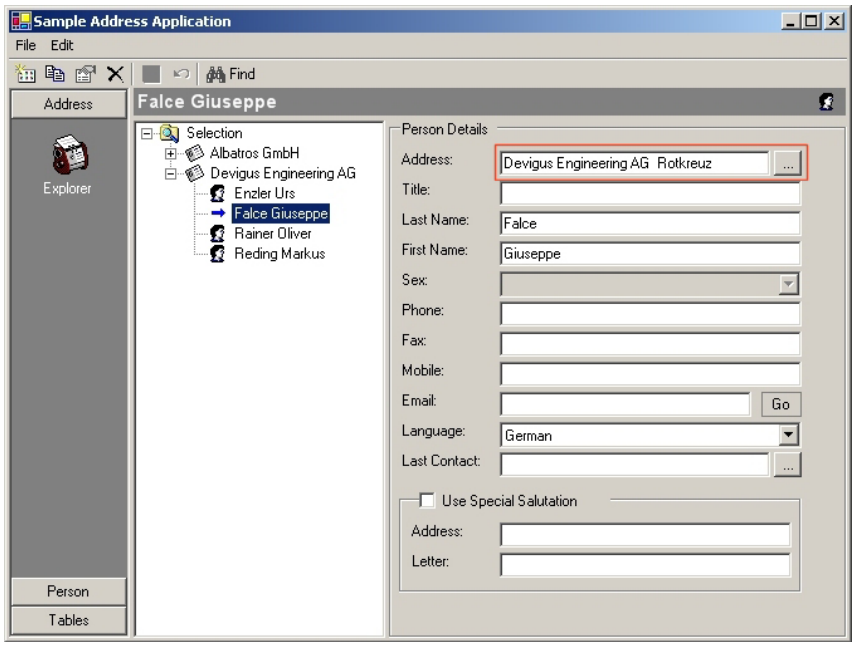


Abbildung 1 Sample Application, Personendetails

Die Abbildung 2 zeigt die Auswahl (Picklist) Funktionalität mit Suchmaske zur Auswahl der entsprechenden Adresse und einem Grid, in welchem die gefundenen Datensätze angezeigt werden. Um eine Adresse auszuwählen, doppelklicken Sie im Grid auf den linken Bereich des Datensatzes (siehe Pfeil in der unteren Abbildung).

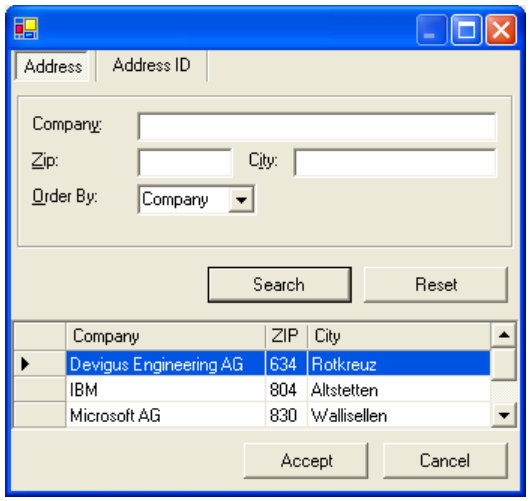


Abbildung 2 PickerDialogAddress

Da sowohl die Klasse *SelectionAddress* als auch *SelectionGridAddress* für die Implementierung des *vdxPickField*-Controls wiederverwendet werden können (exakt dieselben controls werden in der standard Addressdatenverwaltung verwendet!), beschränkt sich der Aufwand auf die nachfolgenden Schritte.

1.1 Definition der Namespaces

Die Platzhalter stehen für die konfigurierten Projektnamen und *Namespaces* im ApplicationWizard.

Platzhalter	Beispiel	Beschreibung
<ClientProjectName>	mySampleClient	Name des Client-Projekts
<ClientProjectNamespace>	mySampleClient	Default-Namespace des Client-Projekts
<WebServiceName>	mySampleWebServices	Name des WebService.

Tabelle 1 Definition der Namespaces

1.2 Erstellen der PickerDialogAddress-Klasse

Folgende Schritte beschreiben das Erstellen der Klasse *PickerDialogAddress* bzw. das Ableiten ihrer Basisklasse *AppPickerDialog*:

1. Selektieren Sie im *Solution Explorer* den Ordner Address.

2. Fügen Sie eine neue, abgeleitete *Control*-Klasse zu ihrem Projekt hinzu. Wählen Sie dazu den Befehl *Add Inherited Form...* aus dem *Project*-Menu aus.

3. Benennen Sie diese Klasse in *PickerDialogAddress* um.

4. Klicken Sie auf *Open* um den Dialog zur Selektion der Basisklasse zu öffnen.

5. Aus der folgenden Komponenten-Liste wählen Sie nun die Klasse *AppPickerDialog* aus und klicken auf den *OK*-Button. Die Abbildung 3 zeigt das Auswahl-Fenster:

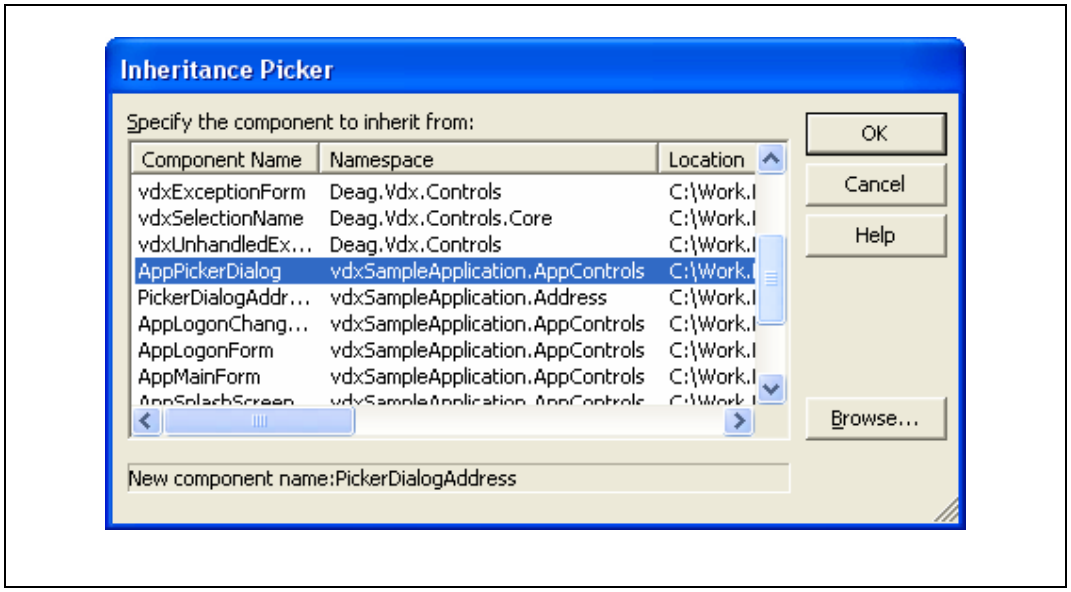


Abbildung 3 Inheritance Form Picker

6. Damit Sie komfortabel auf die VDX-Klassen zugreifen können, fügen Sie die folgenden VDX-Namespace-Referenzen in den Code ein:

```
using System.Data;
using System.Text;
using Deag.Vdx.Common;
using Deag.Vdx.Controls.Core;
using Deag.Vdx.Controls.Standard;
using <ClientProjectNamespace>.<WebServiceName>;
```

Codesegment 1 Using-Anweisungen von PickerDialogAddress

7. Kompilieren Sie die Solution.

8. Setzen Sie im Property-Browser das Property *SearchControlClassName* auf *mySampleClient*.Address.*SelectionAddress*.

9. Analog dazu setzen Sie den NavigatorClassName auf *mySampleClient.Address.SelectionGridAddress.*

1.3 Erstellen der DataPickFieldAddress-Klasse

Folgende Schritte beschreiben das Erstellen der Klasse *DataPickFieldAddress* bzw. das Ableiten ihrer Basisklasse *AppDataPickField*:

1. Fügen Sie dem Ordner Address in ihrem Projekt eine neue, abgeleitete *UserControl*-Klasse hinzu. Wählen Sie dazu den Befehl Add Inherited Control... aus dem Project-Menu aus.

2. Benennen Sie die Klasse in *DataPickFieldAddress* um.

3. Klicken Sie auf *Open* um den Dialog zur Selektion der Basisklasse zu öffnen.

4. Aus der folgenden Komponenten-Liste wählen Sie nun die Klasse *AppDataPickField* aus und klicken auf den *OK*-Button. Die nachfolgende Abbildung zeigt das Auswahl-Fenster:

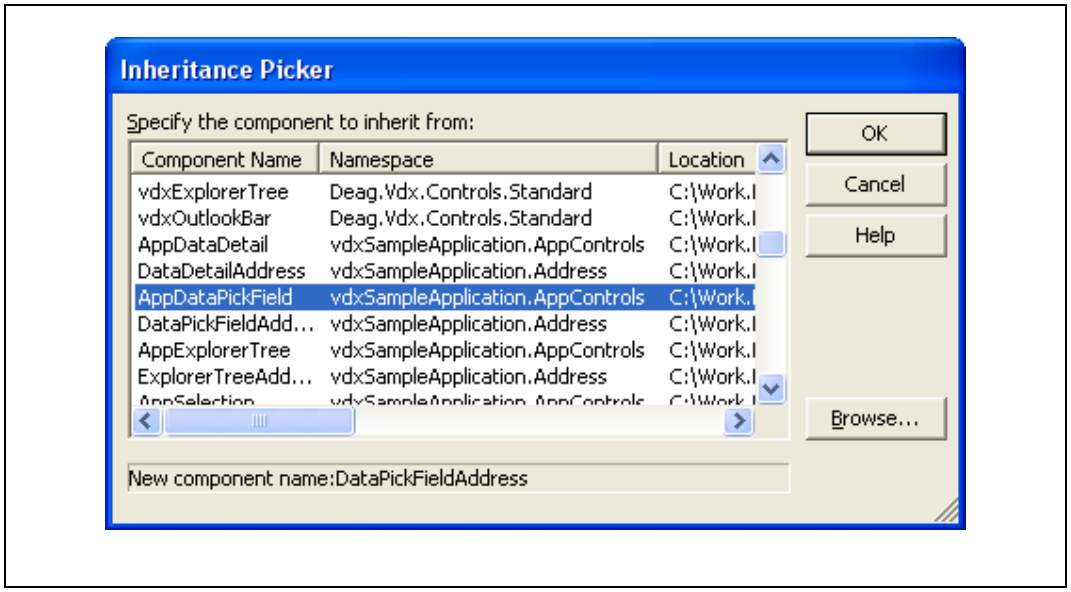


Abbildung 4 Inheritance Control Picker

5. Die neu erstellte Klasse besteht durch die Ableitung bereits aus einem *Code*-Feld, einem *Auswahllisten-Selektor* (Knopf mit drei Punkten), einem Löschen-Button und einem *Description*-Feld.
Standardmässig ist der Löschen-Button nicht sichtbar. Über das Property Visible des Controls *btnClear* kann dieses sichtbar gemacht werden.

Hinweis In der Designer-Ansicht des PickFields sind immer alle Controls sichtbar unabhängig davon, ob das Property *Visible* auf *false* oder *true* gesetzt ist. Erst in der Designer-Ansicht des DataDetails, auf welches das PickField positioniert wird, werden die PickField-Contros nicht mehr dargestellt.

6. Damit Sie komfortabel auf die VDX-Klassen zugreifen zu können, fügen Sie die folgenden VDX-Namespace-Referenzen in den Code ein:

```
using Deag.Vdx.Common;
```

Codesegment 2 Using-Anweisung von DataPickFieldAddress

7. Kompilieren Sie die Solution.

8. Setzen Sie das Property PickerClassName auf *mySampleClient*.Address. DataPickFieldAddress im Property-Browser.

9. Setzen Sie das Property Visible des *txtCode* Controls auf false. Das Code-Feld wird auf diese Weise zur Laufzeit nicht angezeigt. Zur Laufzeit kann deshalb keine Address-Id direkt in das Code-Feld des Pickfields eingegeben und damit keine Adresse ohne Aufrufen der Auswahlliste ausgewählt werden.

1.4 Implementierung der PickerDialogAddress-Klasse

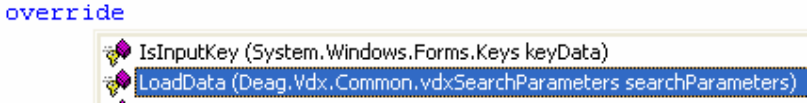
In den folgenden Abschnitten werden die notwendigen Methoden der Basisklasse überschrieben, damit das gewünschte Verhalten entsprechend implementiert werden kann.

1.4.1 Überschreiben der *LoadData*-Methode

Sobald der User auf den *Go*-Button der Suchmaske klickt, muss der entsprechende Datenbezug ausgelöst werden. Der Datenbezug wird – analog zur Klasse *ExplorerAddress* (siehe dazu auch das Tutorial „vdxExplorerTree - Standard Datenmanipulation mit Treeview.“) – in der Methode *LoadData* implementiert. Um diese Funktionalität zu implementieren, führen Sie die nachfolgenden Schritte durch:

1. Überschreiben Sie die *LoadData*-Methode.

Tipp Überschreiben einer Methode über Code-Completion, geben Sie **override** mit nachfolgendem Leerschlag ein und wählen Sie die Methode *LoadData* aus der Liste aus.



2. Fügen Sie den folgenden Code in die Methode ein:

```
return WebServiceProvider.wsAddress.GetDataSetAddressList(
       searchParameters.GetParameters());
```

Codesegment 3 Überschreiben der LoadData-Methode von PickDialogAddress

Hinweis Die Parameterübergabe läuft jeweils über ein *vdxSearchParameters*-Objekt mit dem Namen *SelectionAddress* oder *SelectionAddressId*, je nachdem aus welcher Suchmaske der User die Suche startet. Ein *vdxSearchParameters*-Objekt enthält jeweils ein oder mehrere *vdxSearchParameter*-Objekte, wobei jedes einen spezifischen Suchbegriff enthält. Mit der *Add*-Methode wird ein solches Objekt dem *vdxSearchParameters*-Objekt hinzugefügt.

Für weitere Informationen zur vdxSearchParameter Klasse wird auf die vdxTechRef.chm Hilfedatei verwiesen.

1.4.2 Überschreiben der *GetPickerInfo*-Methode

Damit das vdxPickField den Wert, der im PickerDialog selektiert wurde mitbekommt, muss die Methode *GetPickerInfo* überschrieben werden. Diese liefert das *vdxPickerInfo*-Objekt, welches die Description, ForeignKey und den Code enthält.

1. Überschreiben Sie die *GetPickerInfo*-Methode und fügen Sie den folgenden Code in die Methode ein.

```
dsAddressList.AddressRow addressRow = navigatorInfo.SelectedRow as
       dsAddressList.AddressRow;
if (addressRow!= null)
{
   StringBuilder sb = new StringBuilder();

   if (!Convert.IsDBNull(addressRow.Company))
   {
      sb.Append(addressRow.Company);
      sb.Append(" ");
   }
   if (!Convert.IsDBNull(addressRow.ZIP))
   {
      sb.Append(addressRow.ZIP);
      sb.Append(" ");
   }
   if (!Convert.IsDBNull(addressRow.City))
   {
      sb.Append(addressRow.City);
   }
   return new vdxPickerInfo(sb.ToString(), addressRow.AddressID, null);
}
return null;
```

Codesegment 4 Überschreiben der GetPickerInfo-Methode von PickerDialogAddress

2. Kompilieren Sie die Anwendung und beheben Sie Fehler falls vorhanden. Nun ist das DataPickFieldAddress einsatzbereit.

1.5 Bildverzeichnis

1.6 Tabellenverzeichnis

1.7 Codesegment-Verzeichnis