Visual.NET Extensions – Tutorial

"vdxExplorerList – Simple data manipulation without treeview"

The extensive application development framework for the simple development of Microsoft Visual Studio.NET database applications!

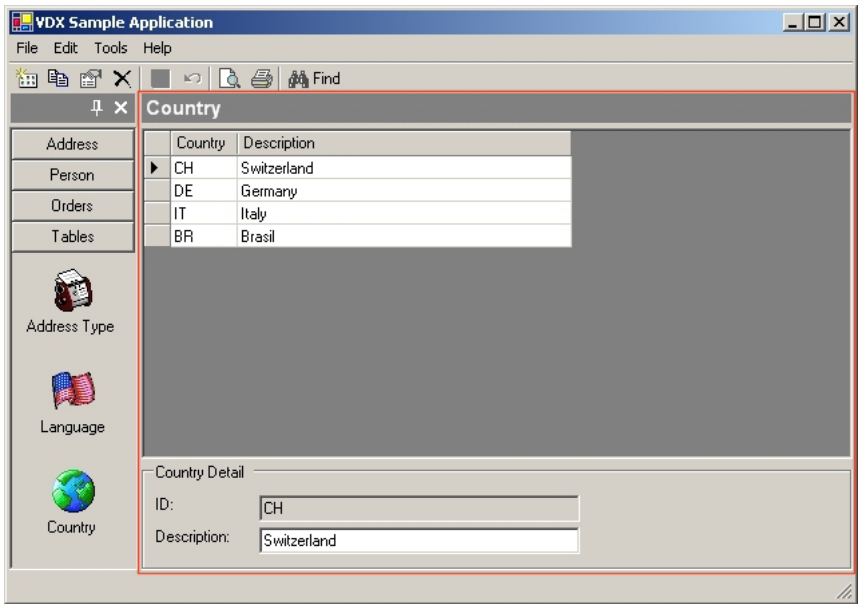
Devigus Engineering AG
Grundstrasse 3
CH-6343 Rotkreuz
Internet: http://www.devigus.com
Email: deag@devigus.com

Version: 1.1
Last Update: 11.11.2002

1 *vdxExplorerList – Simple data manipulation without treeview*

This tutorial describes the usage of the *vdxExplorerList*-control showing a simple data manipulation without treeview. Such a scenario usually gets implemented for simple tables which are referenced by other tables as foreign keys. An example of such a reference table is the country table which gets referenced from the address table.

Following screen shows the *vdxExplorerList*-control with its child-controls *SelectionListCountry* (the list) and the the *DataDetailCountry*-Control (data entry fields below):

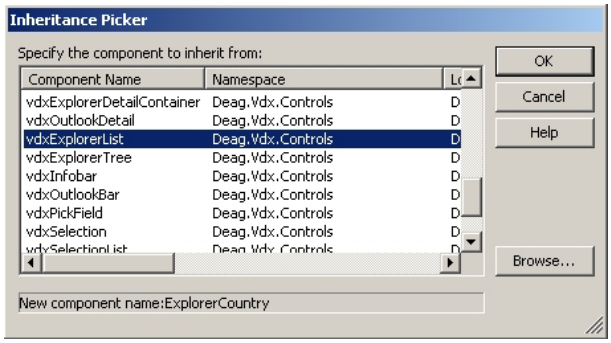


1.1 Creation of the ExplorerCountry-class

This section explains the construction and integration of the countries container class. The database controls for searching and reading countries and also country details will be embedded in this container, similar to the addresses. Different than in the addresses example, the controls for showing the country list and country details will be displayed at the same time and there will be no tree view for navigation.

Following steps describe the creation of the *ExplorerCountry* container-class:

1. Add a new derived *UserControl* class to the project (*Project* → *Add Inherited Control...*).
2. Rename the class *ExplorerCountry*.
3. Click *Open* to open the dialogue box for selecting the base class.
4. Click *Browse...* to supply the appropriate assembly. Select *vdxControls\bin\release\vdxControls.dll* from the VDX installation subdirectory. Select the *vdxExplorerList* class from the component list.



5. Through inheritance, the new class already contains a title bar and a container for the display form.

6. Add the following VDX namespace references in the code to allow easier access to the VDX classes.

```
using Deag.Vdx.Common;
using Deag.Vdx.Common.Enums;
using Deag.Vdx.Controls;
```

The main functions will be added to this class later.

The *ExplorerCountry*-class will serve as a container for the individual country controls (searching countries, displaying the retrieved countries and their details) which will be created in the following sections.

1.2 Creation of the SelectionCountry-class

This section explains the building of the control class which can be used by a user to enter criteria for a search of countries.

1. Add a new derived *UserControl* class to the project (*Project* → *Add Inherited Control...*).

2. Rename the class to *SelectionCountry*.

3. Click *Open* to open the dialog box for selecting the base class.

4. Click *Browse...* to supply the appropriate assembly. Select *vdxControls\bin\release\vdxControls.dll* from the VDX installation subdirectory. Select *vdxSelectionTab* from the component list.

5. Add the following VDX namespace references in the code to allow easier access to the VDX classes.

```
using Deag.Vdx.Common;
using Deag.Vdx.Common.Enums;
using Deag.Vdx.Controls;
```

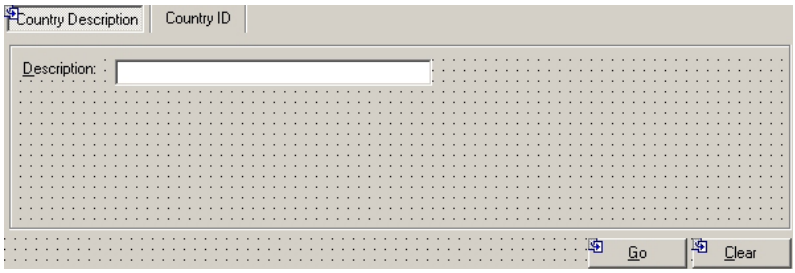
6. The base class already contains a tab control and allows someone to easily design various search forms. The derived control also inherited two buttons: The *Go* button starts the search and the *Clear* button deletes the contents of the text boxes on the control. Select the *vdxTabPages* property and add two *TabPages* to it. Rename the first *TabPages* *tbpCountryDescr* and the second *tbpCountryId*. Set the *Text* and *Name* properties of the two new *TabPages* as shown in the following table:

Added TabPage	Text	Name
vdxTabPage1	Country Description	TbpCountry
vdxTabPage2	Country ID	TbpCountrId

7. You can design the form however you please, so the following are only guidelines. Click the *tbpCountry TabPage* and add a *Label* and a *TextBox* control. Set the *Text* and *Name* properties to the following values.

Added Control	Text	Name
Label	Description	lblCountry
TextBox	""	txtCountry

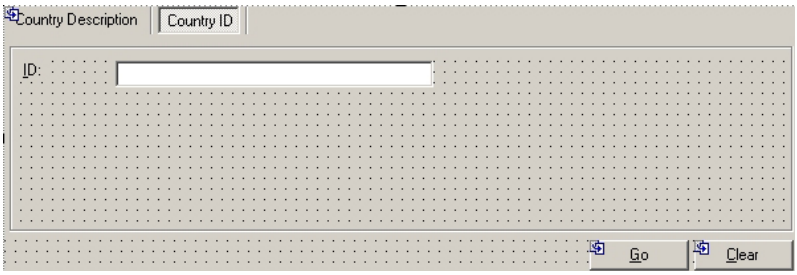
8. Arrange the controls on the form. The following figure shows one suggestion.



9. Click the *Country ID TabPage* and add a *Label* and a *TextBox* control to it. Set the *Text* and *Name* properties of the controls to the following values.

Added Control	Text	Name
Label	ID	LblCountryID
TextBox		txtCountryID

10. Arrange the controls on the *TabPage*. Below is a suggestion.



1.3 Creation of the SelectionListCountry-class

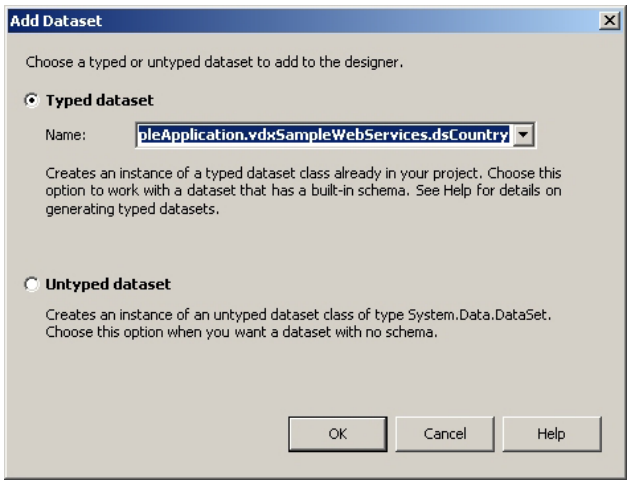
This section describes how to build the control class which displays the found countries to the user in a list.

1. Add a new derived *UserControl* class to the project (*Project* → *Add Inherited Control...*).
2. Rename the class to *SelectionListCountry*.
3. Click *Open* to open the dialog box for selecting the base class.
4. Click *Browse...* to supply the appropriate assembly. Select *vdxControls\bin\release\vdxControls.dll* from the VDX installation subdirectory. Select the *vdxSelectionListDataGrid* class from the component list.

1.3.1 DataSet *dsCountry* and *vdxDataGrid* binding

In order to show the selected countries in the list, we must add a *dsCountry*-DataSet to the *SelectionListCountry*-class and bind it to the *vdxDataGrid*-control. This *DataSet* ist contained in the Server-Part and therefore added trough reference:

1. Select the *SelectionListCountry*-Control in Design-View.
2. On your Toolbox, open the register *Data* und drag a *DataSet* on our control.
3. In the following dialog select the typed *DataSet* *MyVdxSampleApplication.vdxSampleWebServices.dsCountry* from the combobox:



4. Rename the DataSet from *dsCountry1* to *dsCountry*.

5. The *dsCountry*-DataSet must now be bound to the *vdxDataGrid*-Control. Therefore select the properties of the *vdxDataGrid*-Control and set the following values:

Property	Selektion
DataSource	dsCountry
DataMember	Country

NOTE: If you first select the *DataSource*-Property and select the DataSet incl. table (*dsCountry.Country*), the *DataMember*-Property will automatically be set.

Compile the project to check the previous code for errors.

NOTE: If the *"Property or indexer 'System.Windows.Forms.DataGrid. VisibleColumnCount' cannot be assigned to -- it is read-only"* error is displayed by the compiler, then remove the following line of code.

```
this.vdxDataGrid.VisibleColumnCount = 2;
```

Whenever a derived *DataGrid* is changed, Visual Studio .NET changes the read-only properties and these must then be manually removed. Because the Visual Studio .NET Designer also creates duplicate statements, remove the duplicates of the following statements.

```
((System.ComponentModel.ISupportInitialize)(this.vdxDataGrid)).BeginInit();
((System.ComponentModel.ISupportInitialize)(this.vdxDataGrid)).EndInit();
```

1.4 Creation of the DataDetailCountry-class

This control displays the details of a specified country and is activated by clicking on a country in the list (*SelectionListCountry*). Create and integrate the control class:

1. Add a new, derived *UserControl* class to the project (*Project* → *Add Inherited Control...*).

2. Rename the class to *DataDetailCountry*.

3. Click *Open* to open the dialogue box for selecting the base class.

4. Click *Browse...* to supply the appropriate assembly. Select *vdxControls\bin\release\vdxControls.dll* from the VDX installation subdirectory. Select the *vdxDataDetail* class from the component list.

5. Add the following VDX namespace references to the code.

```
using Deag.Vdx.Common;
using Deag.Vdx.Common.Enums;
using Deag.Vdx.Controls;
using Deag.Vdx.Controls.Delegates;
using Deag.Vdx.Controls.EventArguments;
```

6. Select the design view of *DataDetailAddress* and drag the following controls onto it. The VDX variants are on the VDX tab of the toolbox.

Control	Text	Name
vdxLabel	ID:	LblCountryId
vdxLabel	Description:	lblCountryDescr
vdxTextBox (vdxControl!)		TxtCountryId
vdxTextBox (vdxControl)		txtCountryDescr

7. Arrange the controls as shown in the figure below.

1.4.1 DataSet *dsCountry* binding

In order to show the details of the currently selected country in the *DataDetailCountry*-Control, we must add a *dsCountry*-DataSet to the *DataDetailCountry*-class:

1. Open the Register *Data* on your Toolbox and drag a *DataSet* on your control.

2. In the following dialog select the typed *DataSet* *MyVdxSampleApplication.vdxSampleWebServices.dsCountry*.

3. Rename the DataSet from *dsCountry1* to *dsCountry*.

4. The *dsCountry-DataSet* must now be bound to the corresponding *TextBox*-Controls. Following table describes the *TextBox*-bindings:

Control	Property DataBindings.Text
txtCountryId	dsCountry - Country.Country
txtCountryDescr	dsCountry - Country.Descr

Compile the project to check the previous code for errors.

1.4.2 Configuration of the *vdxDataStatusManager*

Because this control will be used to change and store data, the referenced *DataStatusManager* must be configured accordingly.

1. Switch to the Design-View of the *DataDetailCountry*-class. Make sure to select the object *DataDetailCountry* in the *Properties*-Drop-Down-Menu.

2. Set the *vdxDataStatusManager* object's properties in the given order to the values shown in the table below:

Property	Selektion
VdxBindingContainer	DataDetailCountry
VdxDataSet	DsCountry
VdxBusinessObjectClassName	[MyVdxSample…]BoCountrySoap
VdxMainDataTable	dsCountry.Country
VdxUpdateDataSetMethodName	UpdateDataSet

NOTE: If the selection for a specific property does not offer the desired options, recompile your solution. To force the designer to write your changes in the code, change the size of your control and recompile.

At this point, the design and naming tasks have been completed for the controls. Compile the application to check the previous code for errors. In the next chapters, we will combine the different controls and implement the corresponding code.

1.5 Programming of the SelectionCountry-Controls

1.5.1 Overwriting the *VdxFillSearchParameters*-method

In order to send the user-entered search criteria to a *Web service*, they must be "packed" into the appropriate method. The following steps implement this functionality:

1. Select the *Class View* of the project (via *View* → *Class View*).

2. Navigate to the *vdxFillSearchParameters* method in the *vdxSelection* class (*MyVdxSampleApplication* → *SelectionCountry* → *Bases and Interfaces* → *vdxSelectionTab* → *Bases and Interfaces* → *vdxSelection*).

3. Execute the *Add* → *Override* command from the context menu of the *vdxFillSearchParameters* method. This adds an empty method to the *SelectionCountry* class, overwriting the base class'.

4. Add the following code to the method.

```
vdxSearchParameters sp;

if (this.SearchTabControl.VdxSelectedTabPage.Equals(this.tbpCountry))
{
   sp = new vdxSearchParameters("SelectionCountryByDescr");
   sp.Add(new vdxSearchParameter("Descr",this.txtCountryDescr.Text,
      vdxSearchType.Equal));
}
else
{
   sp = new vdxSearchParameters("SelectionCountryById");
   sp.Add(new vdxSearchParameter("Country",this.txtCountry.Text,
      vdxSearchType.Equal));
}

return sp;
```

This method is called when the user clicks the *Go* button on the search control. The parameters are passed by the *vdxSearchParameters* object. The *vdxSearchParameters* object contains one or more *vdxSearchParameter* objects, each containing a specific search criteria. The *Add* method adds such an object to the *vdxSearchParameters* object.

The *vdxSearchParameter*'s constructor requires a name for the search criteria, the search criteria, and the search method.

1.6 Configuration of the ExplorerCountry-Control

To allow the *ExplorerCountry* to use the embedded controls *SelectionCountry*, *SelectionListCountry* and *DataDetailCountry*, following steps are necessary:

1. Open the Design-View of the *ExplorerCountry*-class and set the focus on the *VdxSelectionControlClassName*-Property.

2. Within the corresponding Drop-Down-Menu, select the class *SelectionCountry*.

3. For the *VdxSelectionListControlClassName*-Property select *SelectionListCountry*.

4. Do the same for the *VdxMainDataDetailControlClassName*-Property. Therein select the *DataDetailCountry*-Control. Following table summarizes these steps:

Property	Auswahl
VdxMainDataDetailControlClassName	VdxSampleApplication.DataDetailCountry
VdxSelectionControlClassName	VdxSampleApplication.SelectionCountry
VdxSelectionListControlClassName	VdxSampleApplication.SelectionListCountry

1.7 Programming the ExplorerCountry-Control

The logic of the embedded controls will be implemented centrally in the class *ExplorerCountry*. Following steps describe the implementation of the corresponding functionality:

1.7.1 Overwriting the der *VdxLoadData*-method

When the user presses the *Go*-Button in the selection dialog, the corresponding data loading process must be invoked. All data loading will centrally be implemented in the method *VdxLoadData*. Follow these steps, to implement the data loading:

1. Select *Class View* within the project (Menu *View* → *Class View*).

2. Navigate to the *VdxLoadData*-method; you find this method in the class *vdxOutlookDetail* (*MyVdxSampleApplication* → *ExplorerCountry* → *Bases and Interfaces* → *vdxExplorerTree* → *Bases and Interfaces* → *vdxOutlookDetail*).

3. Select *Add* → *Override* from the context menu of the *VdxLoadData*-method. This adds an empty method in the *ExplorerCountry*-class, overwriting the method from the base class.

4. Add following code:

```
vdxSampleWebServices.BoCountrySoap boCountry = new vdxSampleWebServices.
   BoCountrySoap();

vdxSampleWebServices.dsCountry dsCountry = boCountry.GetDataSetCountry(
   vdxSearchParameters.GetParameters());

DataSet dsRet = null;
if (dsCountry.Tables["Country"].Rows.Count > 0)
{
   dsRet = dsCountry;
}

return dsRet;
```

NOTE: First, a proxy-object of the corresponding *BoCountry-WebService*-class will be created and thereon the *GetDataSetCountry*-method will be called. This fills the DataSet, which will be used to populate the list resp. (in the case of the data detail) the detail control. The content of the DataSets depends from the user entered search parameters (*vdxSearchParameters*). With the *GetParameters*-method, a *vdxSearchParameters*-Object will be serialized for easy transportation to the Server.

For more information regarding the vdxSearchParameters class consult the vdxTechRef.chm helpfile!

1.7.2 Overwriting the *VdxBuildSearchParameters*-method

If the user clicks on a DataRow of the DataGrid, as a preparation for the dataloading for the corresponding data detail control, a *vdxSearchParameters*-object which identifies the DataRow must be build. To implement this functionality, follow these steps:

1. Select the *Class View* of the project.

2. Navigate to the *VdxBuildSearchParameters*-method; this method is located in the class *vdxOutlookDetail* (*MyVdxSampleApplication* → *ExplorerCountry* → *Bases and Interfaces* → *vdxExplorerList* → *Bases and Interfaces* → *vdxOutlookDetail*).

3. Select *Add* → *Override* from the contextmenu of the *VdxBuildSearchParameters*-method. This adds an empty method to the *ExplorerCountry*-class, overwiting the one from the base class.

4. Add the following code to this method:

```
vdxSearchParameters sp = new vdxSearchParameters("Country");

sp.Add(new vdxSearchParameter("Country",((string)dataRow["Country"]),
   vdxSearchType.Equal));

return sp;
```

   The created *vdxSearchParameters*-object will be used in the *VdxLoadData*-method, where the corresponding data loading occurs.

1.7.3 Overwriting the *VdxUpdateSelectionList*-method

In order to update a user modified record also on the list, we must override the method *VdxUpdateSelectionList*:

1. Select the *Class View* of the project.

2. Navigate to the *VdxUpdateSelectionList*-method; you find this method in the class *vdxOutlookDetail*.

3. Select *Add* → *Override* from the contextmenu of the *VdxUpdateSelectionList*-method. This adds an empty method in die *ExplorerCountry*-class, which overrides the one from the base class.

4. Add following code to this method:

   This methode determines the list values of the corresponding *DataRow* and transfers them to the *SelectionList* in the *SelecitonListAddress* class.

```
targetValues = base.VdxDataRowToSelectionListValueArray(sourceDataRow);
base.VdxUpdateSelectionList(targetValues,sourceDataRow);
```

1.8 Programming the vdxActionItem

In order to load the country codes by clicking on the country icon on the Outlookbar, either the corresponding click event must be implemented or the icon must be linked to the *vdxDataActionManager* object. The following steps describe the latter approach.

Select the properties of the *vdxDataActionManager* object on the *MainForm*.

1. Open the collection editor of the *VdxActionItem* properties.

2. Click *Add* and rename the *vdxActionItem* object to *aiCountryExplore*.

3. To link the action event with the corresponding control, select the *iconCountryExplore* entry in the *VdxInvokerControl1* property.

   NOTE: Because one cannot switch to the various events directly from the collection editor, one must close the collection editor and select the *ActionItem* object by selecting it from the ComboBox of the *Properties* window.

4. Select the properties of the *aiCountryExplore* object.

5. Select the events view (button with the yellow lightning bolt) and double-click on the *vdxActionInvoked* event of the *aiCountryExplore* object. This adds the *aiCountryExplore_vdxActionInvoked* event listener method to the *MainForm* class.

6. Add the following code to the method.

```
this.vdxMultiContainer.VdxCurrentControlType =
   typeof(MyVdxSampleApplication.ExplorerCountry);

((vdxOutlookDetail)this.vdxMultiContainer.VdxCurrentControl).Show();
```

   When the country icon on the Outlookbar is clicked, this code will load the appropriate search form using the *SelectionCountry* class. If a selection already exists, then it will be displayed.

Compile and start the application. Following screen shows the completed country reference table management: