

Visual.NET Extensions – Tutorial

“vdxExplorerGrid – Einfache Tabellenverwaltung
ohne Treeview“

**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	<i>vdxExplorerGrid</i> - Einfache Tabellenverwaltung ohne Treeview	3
1.1	Definition der Namespaces.....	3
1.2	Erstellen der ExplorerCountry-Klasse.....	4
1.3	Erstellen der SelectionCountry-Klasse.....	6
1.4	Erstellen der DataDetailCountry-Klasse	9
1.4.1	Konfiguration des <i>dataStatusManager</i>	10
1.4.2	DataStatusManager an Controls binden	10
1.5	Programmierung des SelectionCountry-Controls.....	10
1.5.1	Überschreiben der <i>FillSearchParameters</i> -Methode	10
1.6	Programmierung des DataDetailCountry-Controls	11
1.7	Konfigurieren des ExplorerCountry-Controls	12
1.7.1	DataSetReference <i>dscCountry</i> an <i>dataGrid</i> binden.....	13
1.8	Programmierung des ExplorerCountry-Controls.....	15
1.8.1	Überschreiben der <i>LoadData</i> -Methode.....	15
1.8.2	Überschreiben der <i>GetLoaderInfo</i> -Methode	16
1.9	Programmierung des <i>vdxActionItem</i>	16
1.10	Bildverzeichnis	18
1.11	Tabellenverzeichnis	18
1.12	Codesegment-Verzeichnis	18

1 *vdxEplorerGrid* - Einfache Tabellenverwaltung ohne Treeview

Dieses Tutorial beschreibt den Einsatz des *vdxEplorerGrid*-Controls bei der Implementation einer einfachen Tabellenverwaltung ohne Treeview. Ein solches Szenario dient der einfachen Verwaltung von Tabellen, deren Inhalt durch andere Tabellen referenziert wird, sogenannte Referenztabelle. Ein Beispiel einer solchen Referenztabelle ist die Länder-Tabelle, deren Inhalt einer Adresse zur Auswahl des entsprechenden Landes dient.

Analog zu den übrigen Tutorials wird anhand der Beispiel-Applikation ein konkreter Anwendungsfall implementiert, welcher die Verwendung der *vdxEplorerGrid*-Klasse exemplarisch beschreibt. Folgende Abbildung zeigt das – in diesem Tutorial zu implementierende – ① *vdxEplorerGrid*-Control inklusive seiner Child-Controls ② *SelectionGridCountry* und ③ *DataDetailCountry*:

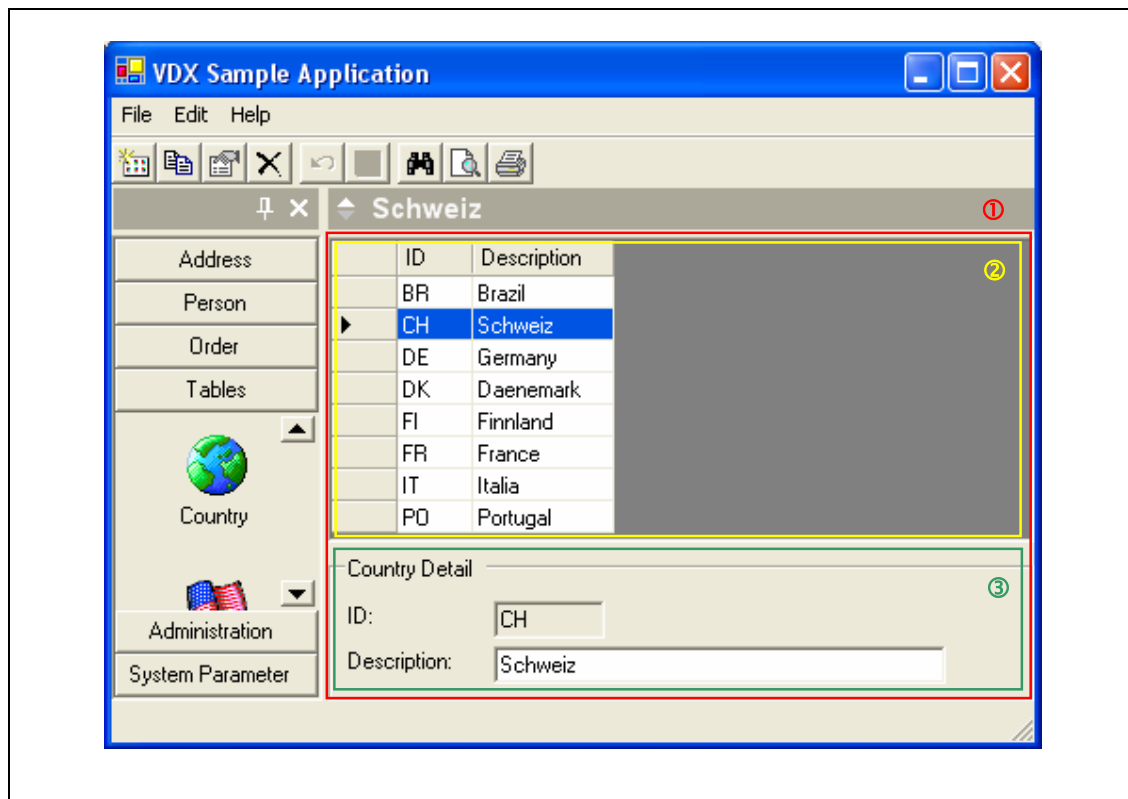


Abbildung 1 Länderverwaltung zur Laufzeit

1.1 Definition der Namespaces

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

Platzhalter	Beispiel	Beschreibung
<ClientProjectName>	mySampleClient	Name des Client-Projektes
<ClientProjectNamespace>	mySampleClient	Default-Namespace für Client-Projekt
<WebServiceName>	mySampleWebServices	Name des WebServices.

Tabelle 1 Definition der Namespaces

1.2 Erstellen der ExplorerCountry-Klasse

In diesem Kapitel wird die Container-Klasse erstellt. In diesem Container werden – analog zum Adressteil im *vdxExplorerTree* – *Standard*-Tutorial – die Ländersuche (*SelectionCountry*), die gefundenen Länder (*SelectionGridCountry*) sowie die Länderdetails (*DataDetailCountry*) eingebettet. Im Unterschied zum Adressteil wird das Detail-Control gleichzeitig mit dem Grid-Control angezeigt und es besteht keine Möglichkeit zur Navigation via Baumansicht bzw. *TreeView*.

Folgende Schritte zeigen das Erstellen der *ExplorerCountry*-Container-Klasse:

1. Erstellen Sie einen neuen Ordner Country.
2. Fügen Sie dem Ordner Country eine neue, von UserControl abgeleitete Klasse hinzu. Wählen Sie dazu den Befehl *Add Inherited Control...* aus dem *Project*-Menu aus.
3. Benennen Sie die Klasse in *ExplorerCountry* um.
4. Klicken Sie auf *Open* um den Dialog zur Selektion der Basisklasse zu öffnen.
5. Im nachfolgenden Dialog wählen sie die Klasse *AppExplorerGrid* als Basis für *ExplorerCountry* aus. Die nachfolgende Abbildung zeigt das Auswahl-Fenster:

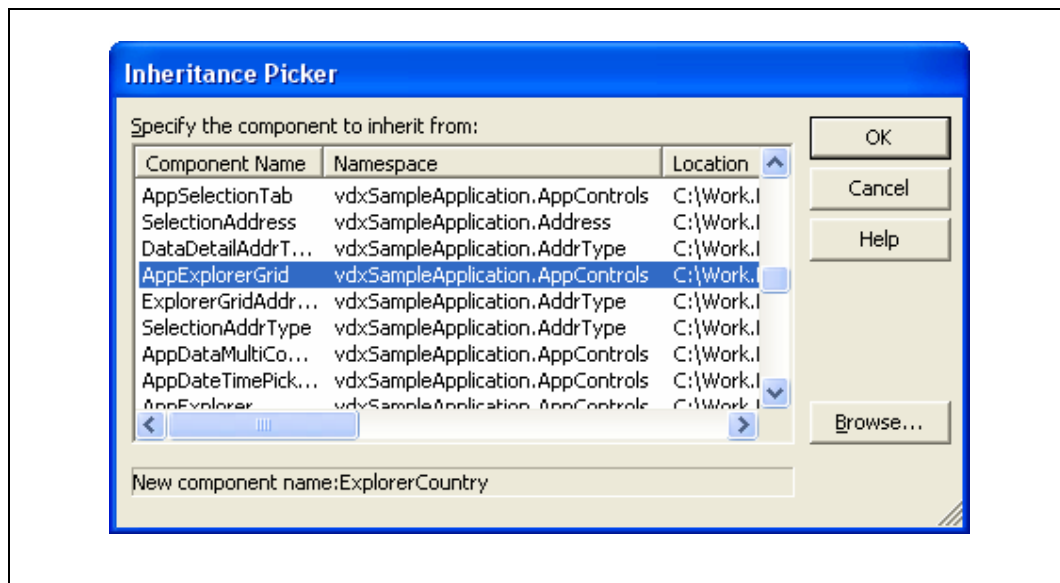


Abbildung 2 Inheritance Picker

6. Die neu erstellte Klasse besteht durch die Ableitung bereits aus einem Container für die entsprechenden Controls (*SelectionCountry*, *SelectionGridCountry* oder *DataDetailCountry*):

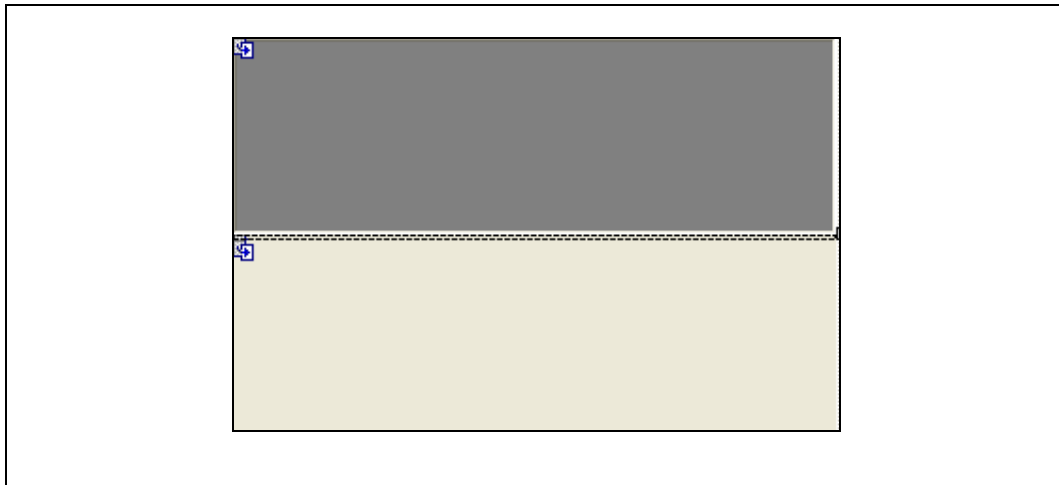


Abbildung 3 Design-Ansicht des ExplorerCountry

7. Wechseln Sie zur Code-Ansicht und kopieren Sie die folgenden Namespace-Referenzen ganz oben in den Code.

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

Codesegment 1 Using-Anweisung von ExplorerCountry

In dieser Klasse wird später die Hauptfunktionalität implementiert. In den folgenden Kapiteln werden zuvor die Klassen *SelectionCountry*, *SelectionGridCountry* und *DataDetailCountry* erstellt.

1.3 Erstellen der SelectionCountry-Klasse

In diesem Kapitel wird die Control-Klasse erstellt, in der ein Benutzer die Suchbegriffe für die Länderreferenzdaten eingeben kann. Folgende Schritte zeigen das Erstellen dieser Control-Klasse:

1. Fügen Sie eine neues *Inherited User Control* SelectionCountry ein, das von AppSelectionTab ableitet.
2. Kopieren Sie folgende VDX-Namespace-Referenzen in den Code.

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

Codesegment 2 Using-Anweisung von SelectionCountry

3. Wechseln Sie zur Design-Ansicht. Passen Sie die Größe des Controls an damit die Buttons „Go“ und „Clear“ sichtbar werden.
4. Die Basisklasse, die bereits ein TabControl enthält, ermöglicht es auf einfache Weise eine differenzierte Suchmaske zu gestalten.

Es gibt zwei Möglichkeiten eine TabPage hinzuzufügen.

- a) entweder direkt *Add Tab* (Abbildung 4) klicken auf dem Property Sheet.
- b) oder über das Property TabPages den *TabPage Collection Editor* (Abbildung 5) öffnen und TabPages hinzufügen

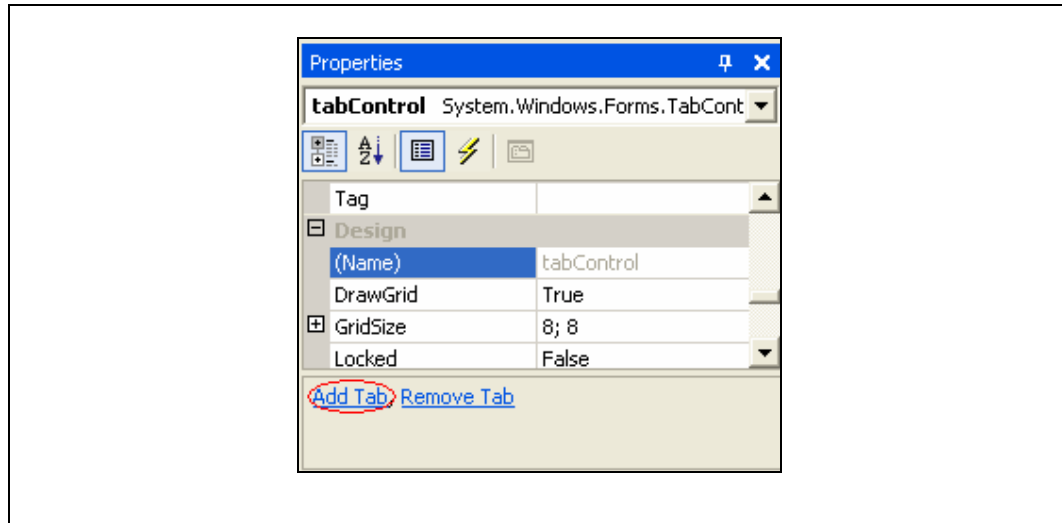


Abbildung 4 Add Tab auf TabControl Property Sheet

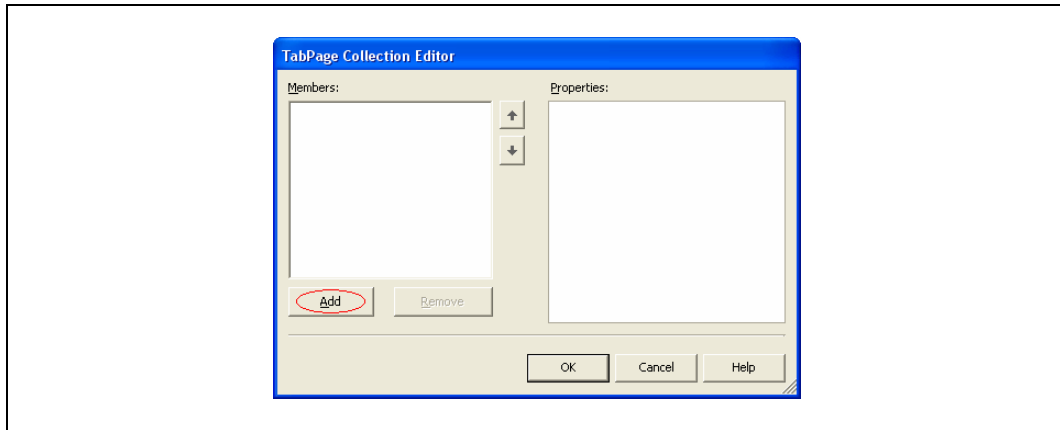


Abbildung 5 TabPage Collection Editor

Fügen Sie zwei *TabPage*s hinzu. Benennen Sie die erste *tbpCountry* und die zweite *tbpCountryId*. Das abgeleitete Control besitzt bereits zwei Buttons. Der *Go*-Button setzt die Suchabfrage ab und der *Clear*-Button löscht alle Inhalte der Textboxen auf diesem Control. Folgende Tabelle enthält das *Text*- und *Name*-Property der zwei neu erstellten *TabPage*s:

Eingefügte TabPage	Text	Name
TabPage1	Country Description	tbpCountry
TabPage2	Country ID	tbpCountryId

Tabelle 2 Konfiguration der SelectionCountry-TabPages

- Grundsätzlich sind Sie bei der Gestaltung der Suchmasken frei. Daher gilt die folgende Gestaltung nur als Richtlinie. Klicken Sie auf *tbpCountry-TabPage* und fügen dort jeweils ein Label und *TextBox*-Control ein. Setzen Sie die *Text*- und *Namen*-Properties der Controls auf folgende Werte:

Eingefügtes Control	Text	Name
AppLabel	Description	lblCountry
AppTextBox		txtCountryDescr

Tabelle 3 Controls auf Country-TabPage

6. Ordnen Sie nun die auf dem Form Controls an. Ein Layout-Vorschlag zeigt die folgende Abbildung:

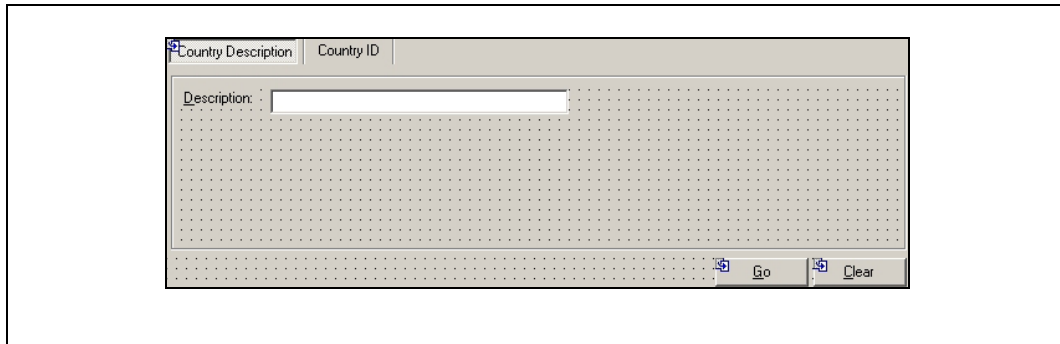


Abbildung 6 SelectionCountry - Tab CountryDescr in Design-Ansicht

7. Klicken Sie nun auf die *Country ID*-TabPage und fügen dort nochmals ein *Label*- und *TextBox*-Control ein. Setzen Sie die Text- und Namen-Properties der Controls auf folgende Werte:

Eingefügtes Control	Text	Name
AppLabel	ID	lblCountryId
AppTextBox		txtCountryId

Tabelle 4 Controls auf ID-TabPage

8. Ordnen Sie die beiden Controls nach Ihrem Wunsch an. Ein Layout-Vorschlag zeigt wiederum die folgende Abbildung:

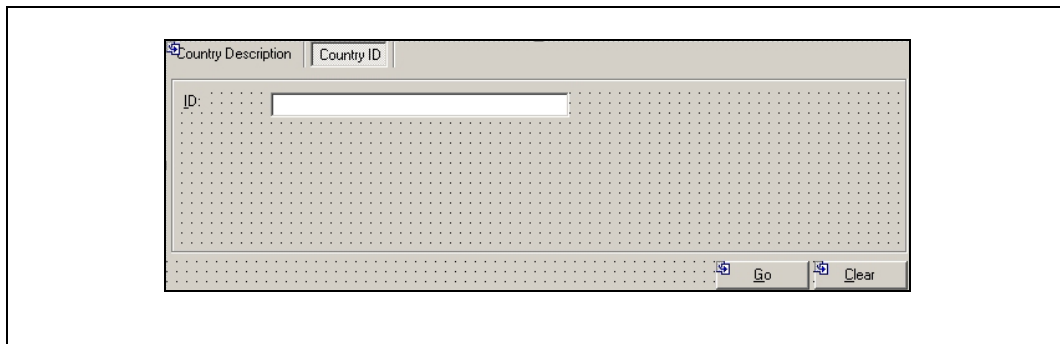


Abbildung 7 SelectionCountry - Tab CountryId in Design-Ansicht

1.4 Erstellen der *DataDetailCountry*-Klasse

Dieses Control präsentiert die Details eines entsprechenden Landes. Die Anzeige kann durch einen einfachen Klick auf ein Land innerhalb der Liste (bzw. des *SelectionGridCountry*-Controls) aktiviert werden. Folgende Schritte zeigen das Erstellen der *DataDetailCountry*-Klasse:

1. Fügen Sie eine neues *Inherited User Control* *DataDetailCountry* ein, das von *AppDataDetail* ableitet.
2. Fügen Sie folgende Namespace-Referenzen zu oberst in den Code ein.

```
using Deag.Vdx.Common;
using Deag.Vdx.Common.Enums;
using Deag.Vdx.Controls.Delegates;
using <ClientProjectNamespace>.<WebServiceName>;
```

Codesegment 3 Using-Anweisung von *DataDetailCountry*

3. Wechseln Sie in die Design-Ansicht der *DataDetailCountry*-Klasse und ziehen folgende Controls darauf; diese befinden sich im *Sample App*-Register der Toolbox:

Control	Text	Name
AppLabel	ID:	lblCountryId
AppLabel	Description:	lblCountryDescr
AppTextBox		txtCountryId
AppTextBox		txtCountryDescr

Tabelle 5 Controls von *DataDetailCountry*

4. Ordnen Sie die Controls wie folgt an:

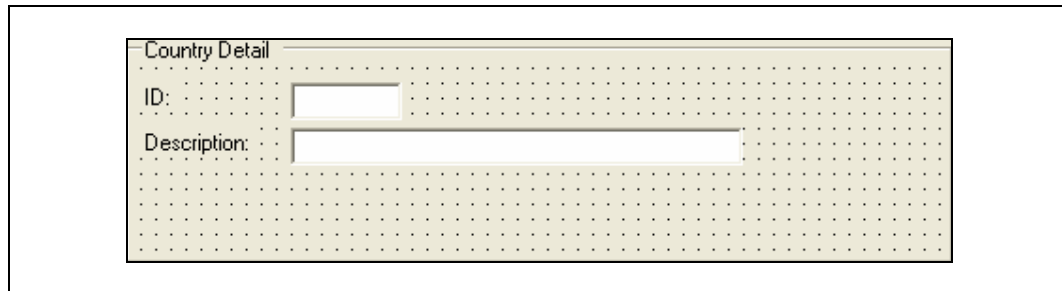


Abbildung 8 *DataDetailCountry* in Design-Ansicht

5. Setzen Sie das *ReadOnly*-Property des *txtCountryId*-Controls auf *true*.
6. Compilieren Sie die Client-Solution erneut.

1.4.1 Konfiguration des *dataStatusManager*

Da im *DataDetailCountry*-Control Daten geändert und gespeichert werden können, müssen Sie das *dataStatusManager*-Control entsprechend konfigurieren:

1. Selektieren Sie *dataStatusManager*-Control und setzen Sie die Properties auf die folgenden Werte damit das DataSet vom DataStatusManager verwaltet werden kann.

Property	Selektion
DataSetType	<ClientProjectNamespace>.<WebServiceName>.dsCountry
MainDataTableName	Country
SelectedDataTableName	Country

Tabelle 6 Konfiguration des DataStatusManager von DataDetailCountry

2. Zusätzlich können auf dem *dataStatusManager* Status-Texte gesetzt werden und Fragestellungen, ob gespeichert oder gelöscht werden soll, formuliert werden.

1.4.2 DataStatusManager an Controls binden

Damit die Details des in der Liste selektierten Landes im entsprechenden *DataDetailCountry*-Control angezeigt werden, ist der *dataStatusManager* an alle nötigen Controls zu binden:

1. Der *dataStatusManager* muss nun an die entsprechenden *TextBox*-Controls gebunden werden. Folgende Tabelle beschreibt die entsprechenden *TextBox*-DataBindingInfos:

Control	Property DataBindingInfos.Text
txtCountryId	dataStatusManager - Country.Country
txtCountryDescr	dataStatusManager - Country.Descr

Tabelle 7 DataStatusManager an Controls binden

2. Kompilieren Sie das Projekt, um den bisher implementierten Code auf Fehler zu überprüfen.

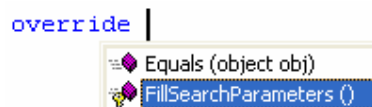
1.5 Programmierung des *SelectionCountry*-Controls

1.5.1 Überschreiben der *FillSearchParameters*-Methode

Um die vom Benutzer eingegebenen Suchbegriffe an den entsprechenden *WebService* zu übermitteln, müssen die Begriffe in der Methode *FillSearchParameters* verarbeitet werden. Um diese Funktionalität zu implementieren, führen Sie die folgenden Schritte aus:

1. Überschreiben Sie die *FillSearchParameters*-Methode

Tipp Überschreiben einer Methode über Code-Completion. Geben Sie **override** mit nachfolgendem Space ein und wählen Sie aus der Liste die Methode *FillSearchParameters* aus.



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

```
vdxSearchParameters sp;  
  
if (this.tabControl.SelectedTab.Equals(this.tbpCountry))  
{  
    sp = new vdxSearchParameters("SelectionCountryByDescr");  
    sp.Add("Descr", this.txtCountryDescr.Text, vdxSearchType.StartWith);  
}  
else  
{  
    sp = new vdxSearchParameters("SelectionCountryById");  
    sp.Add("Country", this.txtCountryId.Text);  
}  
  
return sp;
```

Codesegment 4 Überschreiben der FillSearchParameters-Methode von SelectionCountry

Diese Methode überschreibt die entsprechende Methode der Basisklasse und wird aufgerufen, sobald der Benutzer auf den *Go*-Button des *SelectionCountry*-Controls klickt.

Die Parameterübergabe läuft wiederum über ein *vdxSearchParameters*-Objekt. Ein *vdxSearchParameters*-Objekt enthält jeweils ein oder mehrere *vdxSearchParameter*-Objekte, wobei jedes einen spezifischen Suchparameter enthält. Mit der *Add*-Methode wird ein solches Objekt erzeugt und hinzugefügt. Details zu den Sucharten finden Sie im *vdxExplorerTree*-Tutorial.

1.6 Programmierung des DataDetailCountry-Controls

1. Damit das *txtCountryId*-Control entsprechend dem *dataStatusManager* Status schreibbar oder lesbar gemacht werden kann, müssen Sie den Event *DataStatusChanged* abonnieren. Ergänzen Sie den Konstruktor wie folgt.

```
this.DataStatusManager.DataStatusChanged +=  
    new vdxDataStatusChangedEventHandler(  
        DataStatusManager_DataStatusChanged);  
this.DataStatusManager.Updater = WebServiceProvider.wsCountry;
```

Codesegment 5 Implementation des DataDetailCountry-Konstruktors

Hinweis Um einen Event zu abonnieren kann mit Eingabe des EventHandlers Propertyts direkt mit TAB der Code erzeugt werden.

Beispiel: „*this.DataStatusManager.DataStatusChanged +=*“ eingeben und zweimal TAB drücken.

2. Implementieren Sie die *DataStatusManager_DataStatusChanged*-Methode wie folgt damit die TextBox schreibbar wird, falls der Eintrag neu erstellt wurde.

```
this.txtCountryId.ReadOnly = e.NewState != vdxDataStatus.New;
```

Codesegment 6 Implementation des DataStatusManager-DataStatusChanged-Listeners

3. Überschreiben Sie die *LoadData*-Methode

```
return WebServiceProvider.wsCountry.GetDataSetCountry(
    searchParameters.GetParameters());
```

Codesegment 7 Überschreiben der *LoadData*-Methode von *DataDetailCountry*

4. Überschreiben Sie die *GetInfoBarText*-Methode

```
dsCountry ds = this.DataStatusManager.DataSet as dsCountry;
if (ds != null && ds.Country.Count == 1)
{
    dsCountry.CountryRow row = (dsCountry.CountryRow )ds.Country.Rows[0];
    return row.Descr;
}
return string.Empty;
```

Codesegment 8 Überschreiben der *GetInfoBarText*-Methode von *DataDetailCountry*

5. Kompilieren Sie das Projekt, um den bisher implementierten Code auf Fehler zu überprüfen...

Hinweis Falls trotz Kompilierung nicht das gewünschte Ergebnis erreicht wird, schliessen sie alle Projektfiles und öffnen Sie sie erneut.

1.7 Konfigurieren des *ExplorerCountry*-Controls

Damit die Container-Klasse *ExplorerCountry* auf die eingebetteten Controls *SelectionCountry*, *SelectionGridCountry* und *DataDetailCountry* zugreifen und diese miteinander verknüpfen kann, müssen folgende Schritte ausgeführt werden:

1. Öffnen sie die Design-Ansicht der *ExplorerCountry*-Klasse und selektieren Sie im Property-Sheet das Property *SearchControlType*.
2. Wählen Sie aus dem entsprechenden Drop-Down-Menu die Klasse *SelectionCountry* aus.
3. Damit der Explorer die Suchmaske als erstes anzeigt, setzen Sie den *DisplayMode* auf *Selection*.
4. Folgende Tabelle fasst die zwei Einstellungen nochmals zusammen:

Property	Auswahl
SearchControlType	<ClientProjectNamespace>.Country.SelectionCountry
DisplayMode	Selection

Tabelle 8 Konfiguration von *ExplorerCountry*

1.7.1 DataSetReference *dsrCountry* an *dataGrid* binden

Damit die selektierten Länder in der Liste angezeigt werden, ist ein *dsrCountry*-DataSetReference der *ExplorerCountry*-Klasse hinzuzufügen und an das *dataGrid*-Control zu binden. Diese *DataSetReference*, referenziert das DataSet *dsAddressList*, welches Bestandteil der Server-Anwendung ist:

1. Öffnen Sie das *ExplorerCountry* -Control in der Design-Ansicht.
2. Öffnen Sie das *VDX-Register* der Toolbox und ziehen Sie ein *vdxDatasetReference* auf das *ExplorerCountry* -Control.

Hinweis Wieso wird kein DataSet auf das Control gezogen? Weil in diesem Falle alle DataSets der Webservice-Referenz in der Designer-Ansicht initialisiert werden, dadurch dauert das Laden des Controls extrem lange (Abhängig von der Anzahl DataSets im Webservice). Im Gegensatz dazu initialisiert die *DataSetReference* nur das benötigte DataSet.

3. Öffnen Sie das Property-Sheet und konfigurieren Sie das *vdxDatasetReference*-Objekt wie folgt:

Property	Selektion
Name	dsrCountry
BindingContainer	ExplorerCountry
RefDataSet	<ClientProjectNamespace>.<WebServiceName>.dsCountry

Tabelle 9 Konfiguration des Country-DataSetReference von ExplorerCountry

4. Das *dsrCountry*-Objekt muss nun an das *dataGrid*-Control gebunden werden. Selektieren Sie dazu die Properties des *data*-Controls und setzen die folgenden Werte:

Property	Selektion
DataSource	dsrCountry
DataMember	Country

Tabelle 10 Konfiguration des DataGrids von ExplorerCountry

5. Compilieren Sie die Client-Solution erneut.
6. Erstellen Sie die *ColumnStyles* über *Generate ColumnStyles* des Property-Sheet.

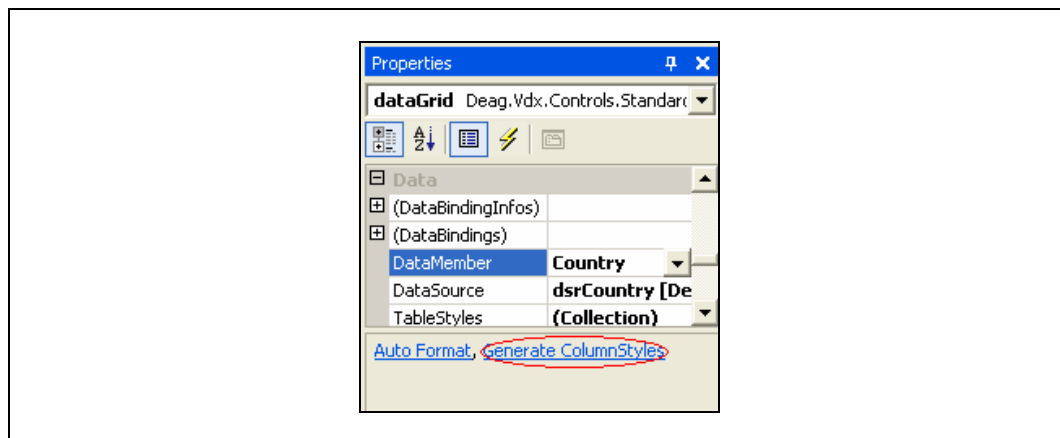


Abbildung 9 ColumnStyles für DataGrid generieren

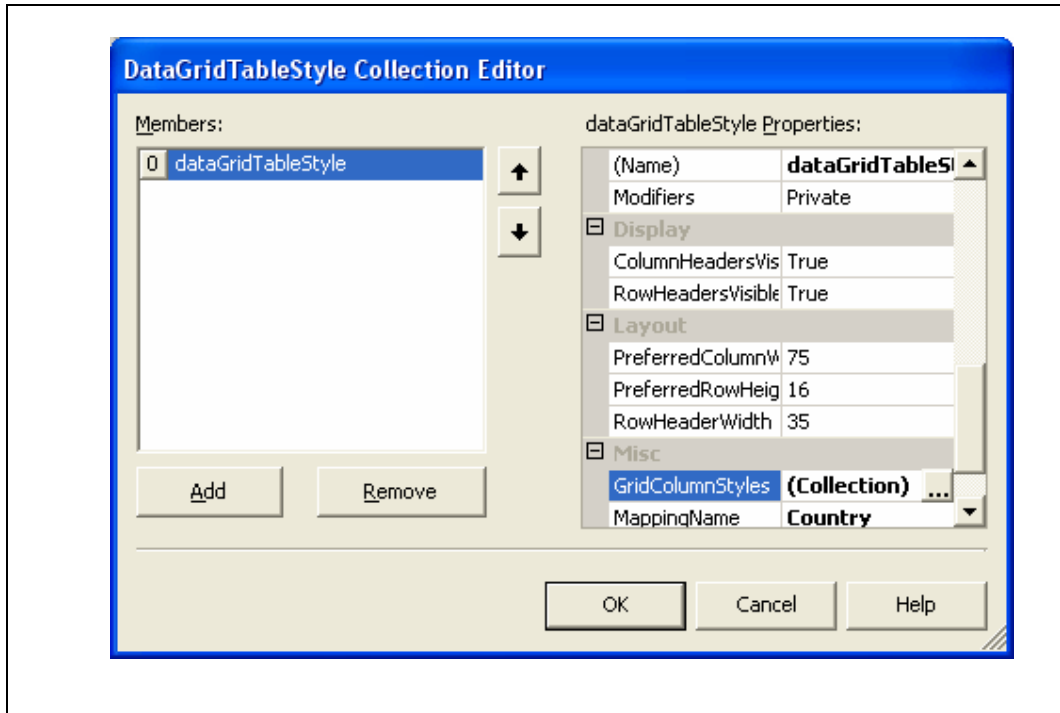


Abbildung 10 DataGridTableStyle Collection Editor

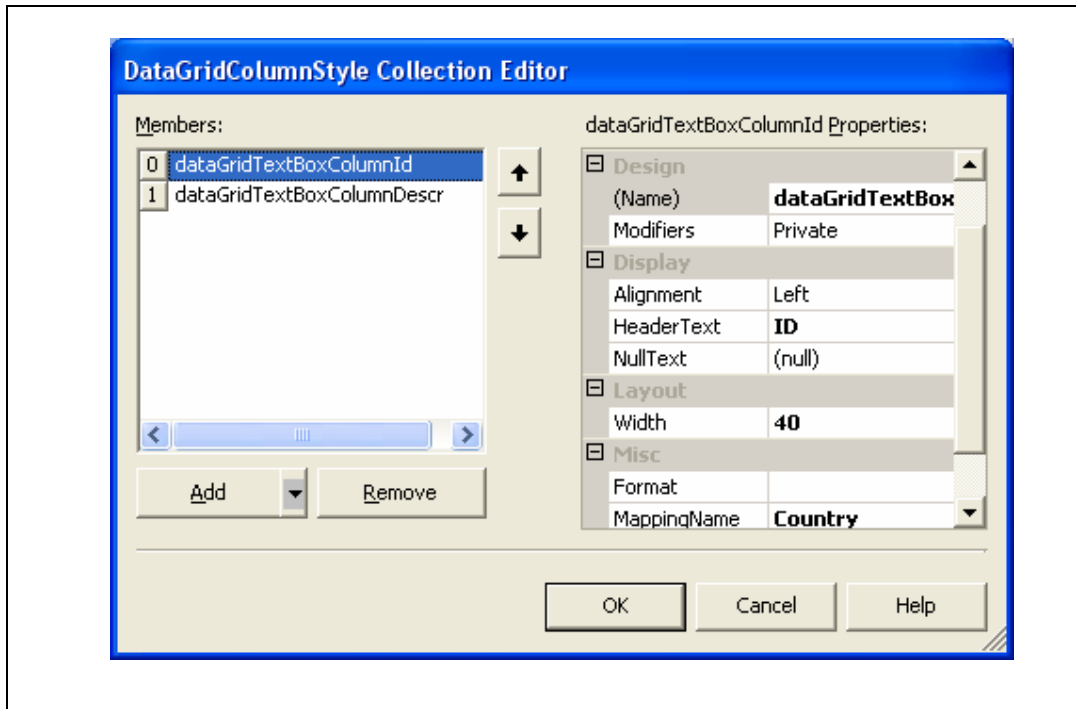


Abbildung 11 DataGridColumnStyle Collection Editor

7. Konfigurieren Sie die ColumnStyles wie folgt:

ColumnStyle	Header Text	Width
dcsCountry	ID	40
dcsDescr	Description	75

Tabelle 11 ColumnStyles des DataGrids von ExplorerCountry

8. Kompilieren Sie das Projekt, um den bisher implementierten Code auf Fehler zu überprüfen.

1.8 Programmierung des ExplorerCountry-Controls

Die Programmlogik der eingebetten Controls wird zentral in der Klasse *ExplorerCountry* implementiert. Folgende Abschnitte beschreiben die Implementation der entsprechenden Funktionalität.

1.8.1 Überschreiben der LoadData-Methode

Sobald der Benutzer auf den *Go*-Button der Suchmaske klickt, soll der entsprechende Datenbezug ausgelöst werden. Der Datenbezug wird – analog zur Klasse *ExplorerAddress* (siehe dazu auch *vdxExplorer-Standard-Tutorial*) – in der Methode *LoadData* implementiert.

1. Überschreiben Sie die *LoadData*-Methode.

```
dsCountry dsCountry = WebServiceProvider.wsCountry.GetDataSetCountry(
    SearchParameters.GetParameters());

if (dsCountry.Country.Count == 0)
{
    MessageBox.Show("No country found. Please try again!",
        "Country Search", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

return dsCountry;
```

Codesegment 9 Überschreiben der LoadData-Methode von ExplorerCountry

Hinweis Zuerst wird ein Proxy-Objekt der entsprechenden *wsCountry-WebService*-Klasse erstellt. Falls noch keines erstellt wurde ist im *WebServiceProvider* die *GetDataSetCountry*-Methode aufzurufen. Diese füllt das *DataSet*, die alle Daten enthält, welche zur Populierung der Liste, bzw. (im Fall des Detaildatenbezuges) des *DataDetail* Controls benötigt werden. Der Inhalt des *DataSet*s ergibt sich abhängig von den durch den Benutzer eingegebenen Suchbegriff (*vdxSearchParameters*). Mit der *GetParameters*-Methode wird ein *vdxSearchParameters*-Objekt serialisiert, damit dieses problemlos auf den Server übertragen werden kann.

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

1.8.2 Überschreiben der *GetLoaderInfo*-Methode

Sobald der Benutzer auf einen Eintrag bzw. eine DataRow des DataGrids klickt, muss als Vorbereitung zum Datenbezug der entsprechenden Detaildaten ein *vdxSearchParameters*-Objekt zwecks Identifikation der DataRow aufgebaut werden.

1. Überschreiben Sie die *GetLoaderInfo*-Methode.

```
vdxSearchParameters sp = null;
dsCountry.CountryRow row =
    navigatorInfo.SelectedRow as dsCountry.CountryRow;

if (row != null)
{
    sp = new vdxSearchParameters("SelectionCountryById");
    sp.Add("Country", row.Country);
}
return new vdxLoaderInfo(sp, typeof(DataDetailCountry));
```

Codesegment 10 Überschreiben der *GetLoaderInfo*-Methode von *ExplorerCountry*

Das erstellte *vdxSearchParameters*-Objekt wird darauf implizit an die *LoadData*-Methode des *DataDetailCountry* übergeben, welche dann den entsprechenden Datenbezug ausführt.

1.9 Programmierung des *vdxActionItem*

Damit die Länderverwaltung durch einen Klick auf das entsprechende Outlookbar-Icon geladen wird, muss entweder der entsprechende Klick-Event implementiert oder das Icon mit dem *actionManager*-Objekt verknüpft werden. Folgende Schritte beschreiben die Verknüpfung mit dem *actionManager*-Objekt:

1. Wählen Sie die Properties des *actionManager*-Objekts; dieses befindet sich auf dem [mySampleClientMainForm](#) (das *actionManager*-Objekt wurde bereits im Rahmen vom ApplicationWizard erstellt).
2. Öffnen Sie den *Collection Editor* des *ActionItems*-Properties.
3. Klicken Sie einmal auf *Add* und benennen Sie das neue *vdxActionItem*-Objekt nach *aiExploreCountry* um.
4. Wählen Sie die Properties des *aiExploreCountry*-Objektes. Beachten Sie dabei, dass im *Collection Editor* nicht direkt auf die entsprechenden Events gewechselt werden kann; daher muss der *Collection Editor* geschlossen und das *vdxActionItem*-Objekt durch Auswahl in der *ComboBox* des *Properties Windows* selektiert werden.
5. Wählen Sie die Events-Ansicht (Schaltfläche mit gelbem Blitz) und Doppel-Klicken auf den *ActionInvoked*-Event des *aiExploreCountry*-Objektes.

6. Dies fügt die *aiExploreCountry_ActionInvoked*-Event-Listener-Methode in die *MainForm*-Klasse ein. Fügen Sie den folgenden Code in diese Methode ein:

```
Show(typeof(Country.ExplorerCountry));
```

Codesegment 11 Implementation des ExploreCountry-ActionItem-Listeners

Dieser Code lädt beim initialen Klick auf das Country-Icon der Outlookbar die entsprechende Suchmaske über die Klasse *SelectionCountry*. Falls bereits eine Selektion existiert, dann wird direkt die letzte Selektion angezeigt.

7. Um das Action-Event mit dem entsprechenden Outlookbar-Icon zu verknüpfen, fügen sie in der Methode *AddInvokers* diese Zeile ein.

```
this.aiExploreCountry.AddInvoker(this.obiExploreCountry);
```

Codesegment 12 Registrierung des ExploreCountry-ActionItems

8. Kompilieren und starten Sie die Anwendung. Folgende Abbildung zeigt die fertiggestellte Länderverwaltung:

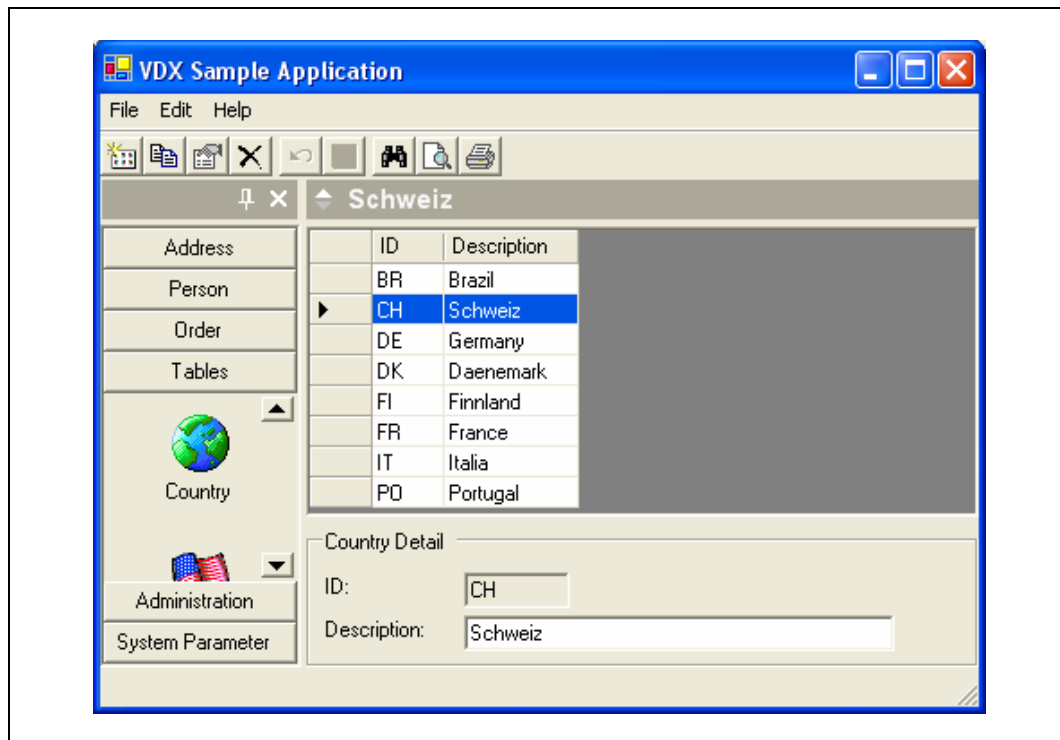


Abbildung 12 Länderverwaltung

Im Tutorial "vdxPickField – Fremdschlüsselvalidierung mit Auswahlliste" wird zusätzlich ein Control erstellt, mit dem man Personen Adressen zuweisen kann; damit können fortan innerhalb der Adressenverwaltung die Personen als Sub-Knoten der Adressen behandelt werden.

1.10 Bildverzeichnis

Abbildung 1 Länderverwaltung zur Laufzeit.....	3
Abbildung 2 Inheritance Picker	4
Abbildung 3 Design-Ansicht des ExplorerCountry	5
Abbildung 4 Add Tab auf TabControl Property Sheet	6
Abbildung 5 tabPage Collection Editor	7
Abbildung 6 SelectionCountry - Tab CountryDescr in Design-Ansicht	8
Abbildung 7 SelectionCountry - Tab CountryId in Design-Ansicht	8
Abbildung 8 DataDetailCountry in Design-Ansicht.....	9
Abbildung 9 ColumnStyles für DataGrid generieren.....	13
Abbildung 10 DataGridTableStyle Collection Editor.....	14
Abbildung 11 DataGridColumnStyle Collection Editor	14
Abbildung 12 Länderverwaltung	17

1.11 Tabellenverzeichnis

Tabelle 1 Definition der Namespaces	3
Tabelle 2 Konfiguration der SelectionCountry-TabPages	7
Tabelle 3 Controls auf Country-TabPage	7
Tabelle 4 Controls auf ID-TabPage	8
Tabelle 5 Controls von DataDetailCountry	9
Tabelle 6 Konfiguration des DataStatusManager von DataDetailCountry	10
Tabelle 7 DataStatusManager an Controls binden.....	10
Tabelle 8 Konfiguration von ExplorerCountry	12
Tabelle 9 Konfiguration des Country-DataSetReference von ExplorerCountry	13
Tabelle 10 Konfiguration des DataGrids von ExplorerCountry	13
Tabelle 11 ColumnStyles des DataGrids von ExplorerCountry	15

1.12 Codesegment-Verzeichnis

Codesegment 1 Using-Anweisung von ExplorerCountry.....	5
Codesegment 2 Using-Anweisung von SelectionCountry	6
Codesegment 3 Using-Anweisung von DataDetailCountry	9
Codesegment 4 Überschreiben der FillSearchParameters-Methode von SelectionCountry	11
Codesegment 5 Implementation des DataDetailCountry-Konstruktors.....	11
Codesegment 6 Implementation des DataStatusManager-DataStatusChanged-Listeners	11
Codesegment 7 Überschreiben der LoadData-Methode von DataDetailCountry	12
Codesegment 8 Überschreiben der GetInfoBarText-Methode von DataDetailCountry	12
Codesegment 9 Überschreiben der LoadData-Methode von ExplorerCountry	15
Codesegment 10 Überschreiben der GetLoaderInfo-Methode von ExplorerCountry	16
Codesegment 11 Registration des ExploreCountry-ActionItems	17
Codesegment 12 Implementation des ExploreCountry-ActionItem-Listeners	17