Wer bin ich?

- Seit 1986 selbständiger Dozent und Entwickler in den Niederlanden
- Kunden sind Behörden und kleine Betriebe
- Viele VBA-Projekte in Office, z.B. Word, Excel und Access-Datenbanken
- Seit 2003 auch Websites mit Datenbank-Anbindung in ASP.NET
- www.birdautomation.nl

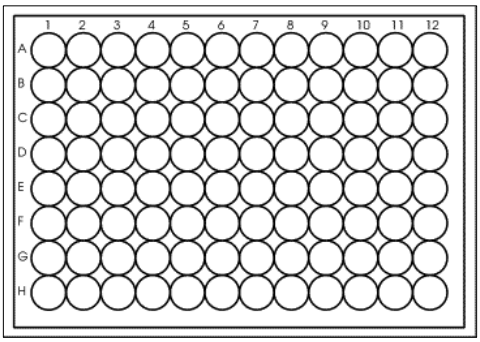
Was möchte ich zeigen?

- Wie es so weit kam
  - Zitate von der IT-Abteilung:
    - "Mit Access möchte man doch nicht mehr gesehen werden…"
    - "Die Datenbanklogik gehört in die Datenbank = SQL Server"
- Classes und Controls
- Datatables, Dataviews und Controls

Was möchte ich vermitteln?

- Programmierung mit Klassen ist abstrakt und aufwändig, aber dafür auch viel flexibler
- .Net hat für fast jedes Problem irgend eine Lösung
  - Jemand anders hat es schon gelöst, man muss es nur finden… (Google)
- Auf Formulare nur wirklich wichtigen, übersichtlichen Code
  - Kein SQL
  - Kein Kampf mit "", &, oder _
- Debugging ist wirklich leicht, wenn man Prozeduren in einem speziellen Testformular erst mal ausprobiert…

Worum geht es…



Theorie

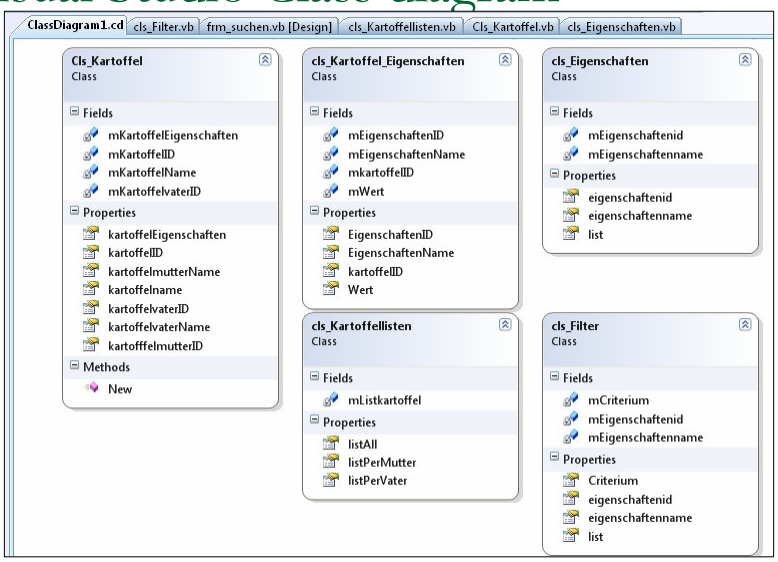
- “In .NET any class or component that implements the IList interface is a valid DataSource. If a component implements the IList interface then it is transformed into an index based collection.

  Some of the classes that support the IList interface in the NET framework are given below. Please note that any class that implements the IList interface is a valid data provider.
- Arrays
- DataColumn
- DataTable
- DataView
- DataSet”

Klassen

- Properties, Methods
- Eine ganz spezielle Property: **List (of T)**

Visual Studio Class diagram



Klasse Beispiel

```
Private mkartoffelID As integer

Public Property kartoffelID() As Integer

        Get
            Return mkartoffelID
        End Get

        Set(ByVal value As Integer)
            mkartoffelID = value
        End Set

End Property
```

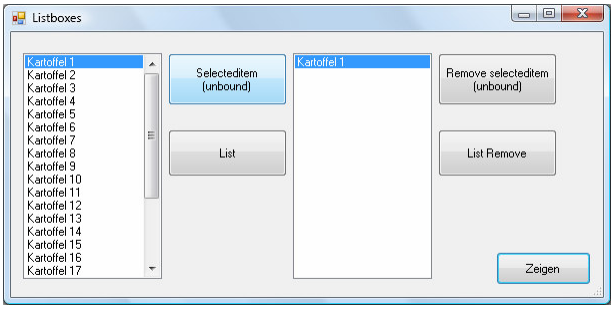
Klasse Beispiel II

```
Private mKartoffelEigenschaften As List(Of
   cls_Kartoffel_Eigenschaften)

Public Property kartoffelEigenschaften() As List(Of
   cls_Kartoffel_Eigenschaften)

   Get
      Return (mkartoffelEigenschaften)
   End Get

   Set(ByVal value As List(Of cls_Kartoffel_Eigenschaften))
       mKartoffelEigenschaften = value
   End Set

End Property
```

Klasse Beispiel III

```
Dim listkartoffel As New List(Of Cls_Kartoffel)
Public ReadOnly Property listAll() As List(Of Cls_Kartoffel)
     Get
         Dim i As Integer
         For i = 1 To 25
                 Dim Cls_Kartoffel As New Cls_Kartoffel
                 Cls_Kartoffel.kartoffelID = i
                 Cls_Kartoffel.kartoffelname = "Kartoffel " & i.ToString
                 Cls_Kartoffel.kartoffelvaterID = Rnd(10) * 10 + 1
                 Cls_Kartoffel.kartofffelmutterID = Rnd(10) * 10 + 1
             REM eigenschaften
             Dim listEigen As New List(Of cls_Kartoffel_Eigenschaften)
             For j = 1 To 5
                 Dim z As New cls_Kartoffel_Eigenschaften
                 z.kartoffelID = i
                 z.EigenschaftenName = "Eigenschaft" & j.ToString
                 z.EigenschaftenID = j
                 z.Wert = Rnd(10) * 10 + 1
                 listEigen.Add(z)
             Next
             Cls_Kartoffel.kartoffelEigenschaften = listEigen
             listkartoffel.Add(Cls_Kartoffel)
         Next
         Return listkartoffel
     End Get
End Property
```

Demo



Eine Listbox an eine Klasse binden

```
Private listKartoffelnAll As List(Of Cls_Kartoffel)

Dim cls_Kartoffellisten As New cls_Kartoffellisten

listKartoffelnAll = cls_Kartoffellisten.listAll

Me.lstAll.DataSource = listKartoffelnAll
Me.lstAll.DisplayMember = "kartoffelname"
Me.lstAll.ValueMember = "kartoffelid"
```

Objekte hin und her schieben (Unbound)

```
Me.lstSelected.DisplayMember = "kartoffelname"
Me.lstSelected.ValueMember = "kartoffelid"


Me.lstSelected.Items.Add(Me.lstAll.SelectedItem)
Me.lstSelected.Refresh()


Me.lstSelected.Items.RemoveAt(Me.lstSelected.SelectedIn
  dex)
Me.lstSelected.Refresh()
```

Objekte hin und her schieben (Bound)

```
If listKartoffelnSelected Is Nothing Then
    listKartoffelnSelected = New List(Of Cls_Kartoffel)
End If

listKartoffelnSelected.Add(Me.lstAll.SelectedItem)

Me.lstSelected.DataSource = Nothing
Me.lstSelected.DataSource = listKartoffelnSelected
Me.lstSelected.DisplayMember = "kartoffelname"
Me.lstSelected.ValueMember = "kartoffelid"

====

listKartoffelnSelected.Remove(Me.lstSelected.SelectedItem)
```

Objekte anzeigen

```
Dim Cls_Kartoffel As Cls_Kartoffel = Me.lstSelected.SelectedItem

MsgBox(Cls_Kartoffel.kartoffelname)

Dim cls_Kartoffel_EigenschaftenList As List(Of
   cls_Kartoffel_Eigenschaften)
Dim cls_Kartoffel_Eigenschaften As cls_Kartoffel_Eigenschaften
cls_Kartoffel_EigenschaftenList = Cls_Kartoffel.kartoffelEigenschaften

For Each cls_Kartoffel_Eigenschaften In cls_Kartoffel_EigenschaftenList
   MsgBox(cls_Kartoffel_Eigenschaften.EigenschaftenName)
Next
```

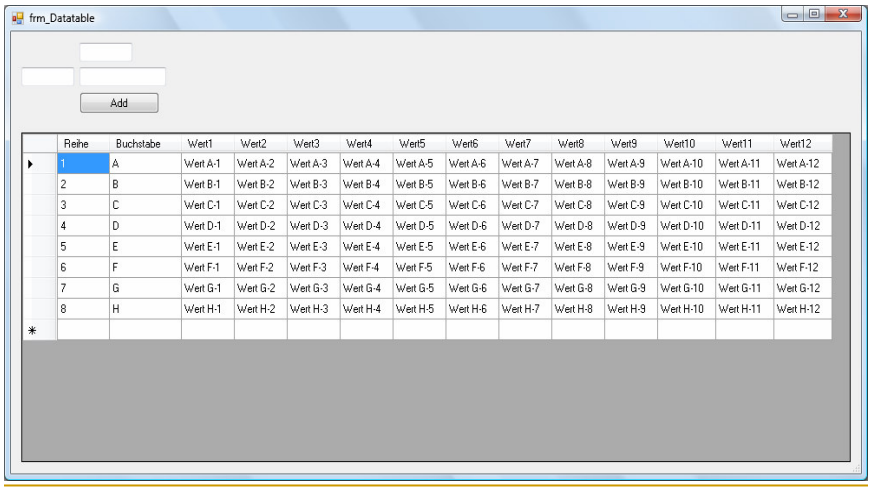
Eine DataGridview an eine Klasse binden

```
Private listKartoffeln As List(Of Cls_Kartoffel)

Private Sub Form1_Load(ByVal sender As Object, ByVal e As
   System.EventArgs) Handles Me.Load
   Dim cls_Kartoffellisten As New cls_Kartoffellisten
   listKartoffeln = cls_Kartoffellisten.listAll

   Me.DataGridView1.DataSource = listKartoffeln

End Sub
```

Datatable

- Virtuelle Tabellen (In Memory)
- Kein SQL
- Sehr flexibel

Beispiel



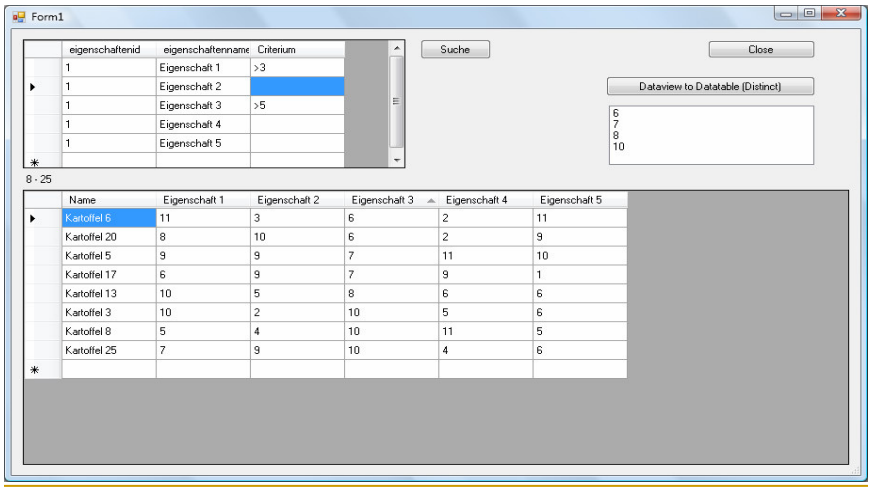
Datatable Beispiel

```
Dim dt As New DataTable()
Dim dr As DataRow
REM Columns
dt.Columns.Add(New DataColumn("Name", GetType(Integer)))
For I = 1 To 12
   dt.Columns.Add(New DataColumn("Wert" & I, GetType(String)))
Next
REM Rows
For I = 1 To 8
   dr = dt.NewRow()
   dr(0) = I
   dr(1) = "Wert 1-" & I.ToString REM
   dr.Item("Wert2") = "Wert 2-" & I.ToString
   For j = 2 To 12
       dr(j) = "Wert " & I.ToString & "-" & j.ToString
   Next
   dt.Rows.Add(dr)
Next
Me.DataGridView1.DataSource = dt
```

Zellen/Koordinaten ansteuern

- Innerhalb einer Datarow (dr)
  - dr(0)
  - dr.Item("Wert2")
- Innerhalb einer DataTable
  - Dt(2)(4) = 3. Zeile, 5. Spalte
  - dt(CType(Me.txtspalte.Text, Integer))(CType(Me.txtZeile.Text, Integer)) = Me.txttext.Text

Demo Dataviews und Datatable



Von Datatable zu Dataviews

- View einer Datatable
  - Sort
  - Rowfilter

```
Dim dvKunden = New DataView(dtKunden)
dvKunden.Sort = "Stadt,Name,Vorname"

Dim strFilter As String = "Feld in (2,3)"
strFilter = "wert >=5 and wert<=9"
Dim dv As DataView = New DataView(dt)
dv.RowFilter = strFilter
```

Von Dataview zurück zu Datatable

```
Dim dtUniekeTraynummers As DataTable = dv.ToTable

Dim dtUniekeTraynummers As DataTable = dv.ToTable(True,
   "nummer")

====

dv.Sort = "[Eigenschaft 3]"
Dim dtdistinct As DataTable = dv.ToTable(True, "Eigenschaft
   3")
Dim drdistinct As DataRow = Nothing
Me.lstdistinct.Items.Clear()
For Each drdistinct In dtdistinct.Rows
    Me.lstdistinct.Items.Add(drdistinct(0))
Next
```

Filterschirm in der Praxis

- Erzeuge in SQL eine Hilfstabelle mit positiven Werten (Tray, Spalte, Zeile)
- Pro distinct Tray wird dann eine neue leere Datatable mit 96 Wells generiert
- Aus der SQL Tabelle wird dann jede positive Zeile und Spalte gelesen und als X und Y verwendet, um den Wert in der DataTable zu platzieren:
  - Dt(intRow)(intCol) = “JA!”

Noch schöner ist BindingList

```
Private WithEvents listOfParts As BindingList(Of Part)
Private Sub InitializeListOfParts()
  ' Create the new BindingList of Part type.
  listOfParts = New BindingList(Of Part)
  ' Allow new parts to be added, but not removed once
    committed.
  listOfParts.AllowNew = True
  listOfParts.AllowRemove = False
  ' Raise ListChanged events when new parts are added.
  listOfParts.RaiseListChangedEvents = True
  ' Do not allow parts to be edited.
  listOfParts.AllowEdit = False
  ' Add a couple of parts to the list.
  listOfParts.Add(New Part("Widget", 1234))
  listOfParts.Add(New Part("Gadget", 5647))
End Sub
```

Endergebnis

- Eine Anwendung, die in jede Richtung erweiterbar ist
- Code ist in kleine Einheiten aufgeteilt
  - Manchmal sucht man, wo sich der Code genau befindet..
    - Rechts-Click -> Go To Definition
  - Eine Lösung besteht aus einer Kette von Teillösungen: man programmiert für jedes Problem eine eigene Methode und verknüpft sie

Tipp

- Einfach diesen Handschuh aufnehmen und selber ausprobieren!!!

Links

- Object Binding Tips and Tricks
  - http://www.code-magazine.com/article.aspx?quickid=0603011
- Windows Forms Object Data Binding in .NET
  - http://www.15seconds.com/issue/040614.htm
- Bindinglist
  - http://msdn.microsoft.com/en-us/library/ms132679.aspx

Class to datatable

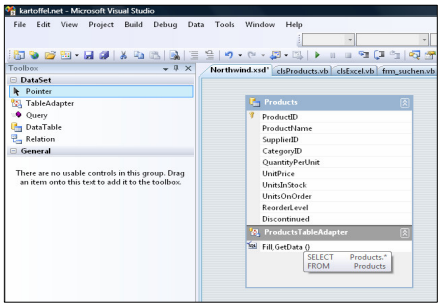
```
Dim dt As New DataTable()
dt.Columns.Add(New DataColumn("Name", GetType(String)))
REM alle Eigenschaftswerte pro kartoffel in datatable aufnehmen als spalte
For Each cls_Eigenschaften In cls_Eigenschaften.list
   dt.Columns.Add(New DataColumn(cls_Eigenschaften.eigenschaftenname,
   GetType(Integer)))
Next
REM data = KartoffelObjektList to datatable
For Each cls_Kartoffel In listKartoffeln
REM neue zeile pro Kartoffel
dr = dt.NewRow()
REM erste spalte = Name
dr("Name") = cls_Kartoffel.kartoffelname
REM jede eigenschaft eine eigene spalte
Dim intZaehler As Integer = 1
For Each cls_Kartoffel_Eigenschaften In
   cls_Kartoffel.kartoffelEigenschaften
   dr(intZaehler) = cls_Kartoffel_Eigenschaften.Wert
   intZaehler = intZaehler + 1
Next
REM zeile an tabelle zufuegen
dt.Rows.Add(dr)
Next
```

Datatabel to Excel

```
Sub DataTableToExcelSheet(ByVal dt As DataTable, ByVal objSheet As
   Excel.Worksheet, ByVal nStartRow As Integer, ByVal nStartCol As
   Integer)
Dim nRow As Integer, nCol As Integer
Try
For nCol = 1 To dt.Columns.Count
      objSheet.Cells(nStartRow, nCol) = dt.Columns(nCol - 1).ColumnName
Next

nStartRow = nStartRow + 1
For nRow = 0 To dt.Rows.Count - 1
  For nCol = 0 To dt.Columns.Count - 1
  If Left(dt.Rows(nRow).Item(nCol), 1) <> "=" Then
        objSheet.Cells(nStartRow + nRow, nStartCol + nCol) =
   dt.Rows(nRow).Item(nCol)
  Else
       objSheet.HPageBreaks.Add(Before:=objSheet.Cells(nStartRow + nRow +
   1, nStartCol + nCol))
  End If
  Next nCol
Next nRow
End Sub
```

Demo Anbindung an Datenbank I

- Dataset Northwind.XSD anlegen



Demo Anbindung an Datenbank II

```
Dim taProducts As New NorthwindTableAdapters.ProductsTableAdapter
Dim tblProducts As Northwind.ProductsDataTable = Nothing
Dim rowProducts As Northwind.ProductsRow = Nothing

REM bitte connectionstring anpassen in app.config
    '<add name="kartoffel.net.My.MySettings.NorthwindConnectionString"
    'connectionString="Data Source=SQLSERVERNAME;Initial
   Catalog=Northwind;Integrated Security=True"
    'providerName="System.Data.SqlClient" />

Public ReadOnly Property productsALL() As List(Of clsProducts)
     Get
         Dim lstProducts As New List(Of clsProducts)
         tblProducts = taProducts.GetData
         For Each Me.rowProducts In Me.tblProducts
             Dim clsProductsDummy As New clsProducts
             clsProductsDummy.productid = rowProducts.ProductID
             clsProductsDummy.productname = rowProducts.ProductName
             REM undsw
             lstProducts.Add(clsProductsDummy)
         Next
         Return lstProducts
     End Get
End Property
```