FB 4 – Wirtschaftsinformatik
Prof. Dr. Peter Zschockelt

Hochschule für Technik
und Wirtschaft Berlin
University of Applied Sciences

Kurzreferenz Arbeit mit LINQ und Stored Procedures unter SQL-Server 2008

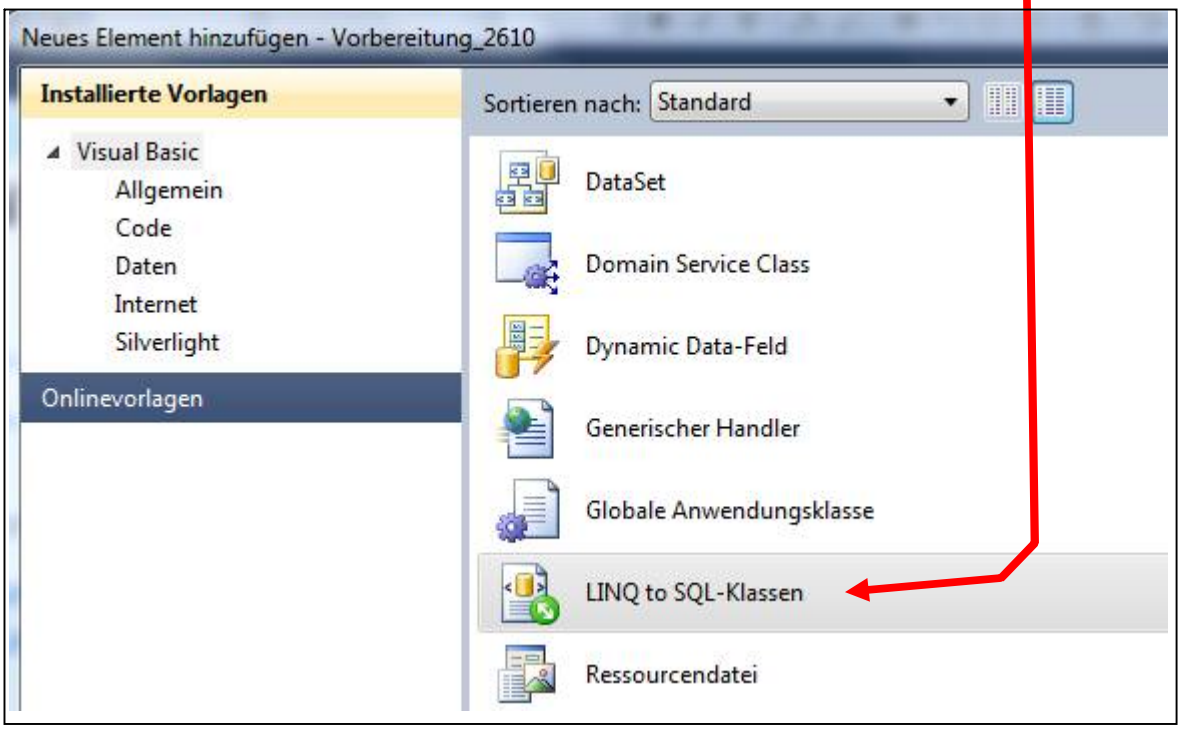
Die Kurzreferenz soll Ihnen die Einbindung von Datenbankdaten aus gespeicherten Prozeduren im SQL-Server 2008 in ASP.NET- Applikationen mittels LINQ am Beispiel verdeutlichen.

Gespeicherte Prozeduren (Stored Procedures) dienen zur Speicherung funktionaler Operationen in der Datenbank. Stored Procedures (SP) werden zu einem beliebigen Zeitpunkt definiert (CREATE PROCEDURE ...) und sind dann für einen beliebigen Aufruf verfügbar.

LINQ (Language Integrated Query) ist eine Komponente des Microsoft .NET-Framework zur Arbeit mit Datenquellen wie zum Beispiel Datenbanken oder XML-Dateien. Teil des umfassenderen LINQ-Konzeptes ist unter dem Begriff "**LINQ to SQL-Klassen**" ein Objekt-Relationales Mapping von Daten aus SP auf Methoden von ASP.NET-Objekten. Damit lassen sich typische Datenbankaufgaben zum Teil erheblich vereinfachen.

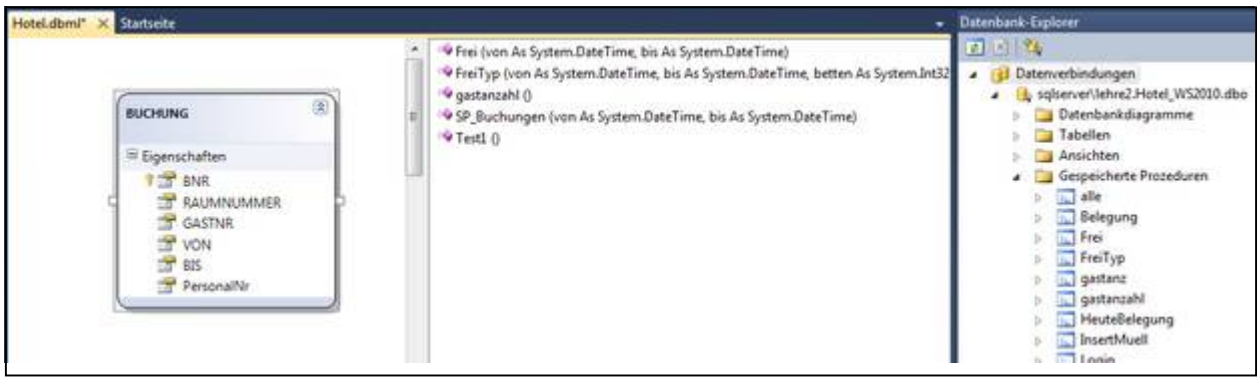
1. Erstellen einer LINQ to SQL Klasse

Fügen Sie Ihrer Projektmappe ein neues Element LINQ to SQL-Klassen hinzu.

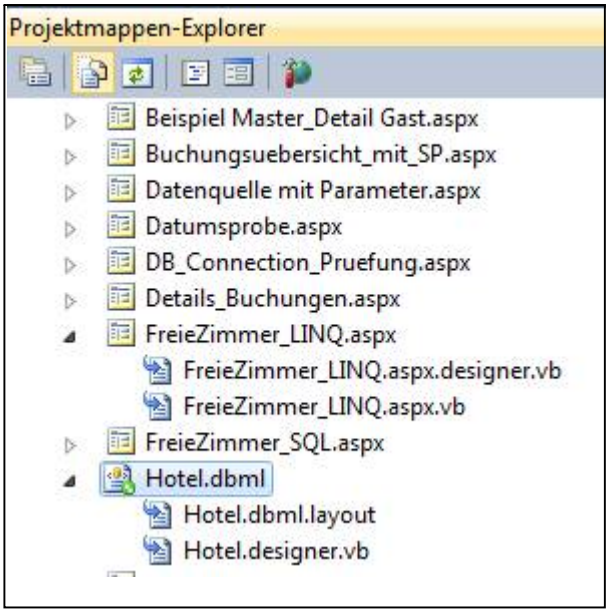


Wählen Sie einen verständlichen Namen für die Klasse, für eine Beispieldatenbank HOTEL z. B. Hotel.dbml und klicken Sie die Taste **Hinzufügen**.

In dem sich öffnenden Fenster (genannt **O/R-Designer**) mit dem Namen der hinzugefügten LINQ to SQL Klasse können Sie aus dem Datenbankexplorer die gewünschten SP in den rechten Teil des Fensters und ggf. auch Tabellen und Views in den linken Teil des Fensters ziehen.



Im Ergebnis erhalten Sie eine automatisch generierte Klasse, die Sie im Projektmappenfenster finden.



In der generierten Klasse wird ein DataContext-Objekt erstellt, welches die Grundlage zur Nutzung aller eingebundenen SP als Methoden des Objekts bildet. Der Name des DataContext-Objektes ergibt sich aus dem Namen der generierten Klasse mit dem Zusatz *DataContext* (z. B. HotelDataContext).

Hinweis: *Prüfen Sie in dem zur Klasse ….dbml gehörigen Fenster* ….designer.vb (z. B. Hotel.designer.vb), ob die von Ihnen ausgewählten Prozeduren über den Rückgabewerttyp ISingleResult verfügen
(z. B. Return CType(result.ReturnValue,ISingleResult(Of FreiTypResult))).
Ansonsten steht Ihnen die Eigenschaft ToList nicht zur Verfügung.

2. Stored Procedures unter VB als Methoden nutzen

Die in der generierten .dbml-Klasse eingebundenen Stored Procedures können als Methoden des DataContext-Objektes genutzt werden. Dazu muss im Programmtext des entsprechenden Webforms zunächst eine neue Instanz des Objektes vereinbart werden.

Beispiel:
```
Dim MeinContext As HotelDataContext = New HotelDataContext()
```

In der Folge kann auf jede Methode im Programmtext zugegriffen werden. Die Parameter der SP – Aufrufparameter ebenso wie Rückgabeparameter – werden als Parameter der Methode angegeben. Der zusätzlich von jeder SP bereitgestellte Rückgabewert kann über die Eigenschaft **ReturnValue** gelesen werden.

Folgendes Beispiel soll dies demonstrieren: Eine SP mit Namen **gastanzahl** soll bei Klicken eines Buttons **Button1** (Ereignisprozedur Button1_Click den Namen des letzten hinzugefügten Gastes als

Rückgabeparameter und die Anzahl der Gäste als Rückgabewert bereitstellen. Im Programmtext der ASP.NET-Applikation könnte dies wie folgt realisiert werden:

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles Button1.Click
        Dim MeineGaeste As New HotelDataContext
        Dim Rparam As String
        MeineGaeste.gastanzahl(Rparam)
        Dim ganz As String = MeineGaeste.gastanzahl.ReturnValue
        TextBox1.Text = "Rückgabewert(Anzahl Gäste) = " & ganz & _
                        " / Rückgabeparameter(letzter Gast) = " & Rparam
End Sub
```

3. Stored Procedures unter VB als LINQ-Datenquelle nutzen

Als LINQ-Datenquelle können sowohl durch den O/R-Designer eingebundene Tabellen und Views als auch eingebundene Stored Procedures genutzt werden. Letztere allerdings nur, wenn sie über das Attribut ToList (vergleiche Hinweis im Abschnitt 1) verfügen.

Eine LINQ-Datenquelle auf Basis einer eingebunden Tabelle oder View kann analog einer SQL-Datenquelle z. B. als Datenquelle einer Gridview über das Fenster GridView-Aufgaben genutzt werden, ohne dass dazu VB-Programmcode erforderlich ist. Allerdings erzielt man damit keinen wesentlichen Qualitätsgewinn gegenüber einer SQL-Datenquelle.

Für die Nutzung einer SP als Datenquelle, die auf Grund der generellen erweiterten Eigenschaften von SP gegenüber einer Tabellen- bzw. View-Datenquelle durchaus von Vorteil sein kann, ist eine programmierte Unterstützung erforderlich. Zunächst aber wählen Sie als Datasource für ein datengebundenes Webcontrol (z. B. eine Gridview) eine LinqDatasource.

Diese Datenquelle brauchen Sie nicht konfigurieren, sondern lediglich im Quellen-Fenster der .aspx-Datei das Attribut "**OnSelecting**" setzen, z. B.

```
<asp:LinqDataSource ID="LinqDataSource1" runat="server"
   OnSelecting ="LinqDataSource_Selecting">
  </asp:LinqDataSource>
```

Die eigentliche Konfiguration der LINQ-Datenquelle erfolgt dann im Programmtext der Applikation. Hier müssen Sie in einer Routine, die auf das Ereignis **OnSelecting** reagiert, einen eigenen Handler für die Linq-Datasource nach folgendem Beispiel programmieren:

Vorausgesetzt wird im Beispiel eine SP Frei(von, bis), mit zwei Aufrufparametern vom Typ Date, welche die Zeitspanne (von – bis) angeben, für die alle freien Hotelzimmer ermittelt werden sollen.

Zugleich soll als Rückgabewert die Anzahl der freien Zimmer im Zeitraum berechnet werden. Die Datumsparameter sollen in zwei Textboxen (TextBox1 und TextBox2) bereitgestellt werden. Die Anzahl der freien Zimmer soll in einem Label (Label1) angezeigt werden.

```
Protected Sub LinqDataSource_Selecting(ByVal sender As Object, _
    ByVal e As LinqDataSourceSelectEventArgs)
  Dim MeinContext As HotelDataContext = New HotelDataContext()
  e.Result = MeinContext.Frei(TextBox1.Text, TextBox2.Text).ToList
  Label1.Text = MeinContext.Frei(TextBox1.Text, TextBox2.Text).ReturnValue
End Sub
```

Das Ereignis **OnSelecting** wird beim ersten Laden der Web-Seite automatisch ausgelöst. Um auf veränderte Datumsparameter in den Textboxen reagieren zu können, genügt ein beliebiges Ereignis, welches eine neue Datenbindung der Gridview auslöst. Das kann beispielsweise ein Seitenwechsel oder eine neue Sortierung der Gridview sein. Eindeutiger für den Nutzer ist jedoch in der Regel eine Taste, die auf das Button_Click–Ereignis wie folgt reagiert:

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles Button1.Click
    'Datasource_Selecting auslösen
    GridView1.DataBind()
End Sub
```