TSQL Mod – Eine DLL für schnelle SQL Server arbeiten

TSQLmod Download

Klasse db (Connection String)

benötigt in erster Linie die SQL Instanz. Nachdem das Objekt erfolgreich initialisiert wurde, wird auch gleichzeitig die Verbindung aufgemacht und die folgenden Methoden können genutzt werden.

LookUP(…)

gibt aus einem SQL Query den ersten Treffer der angegebenen Spalte als String wieder. Ideal um einen Wert aus der Datenbank auszulesen. Möglich ist es entweder die Spaltennummer oder den Spaltennamen anzugeben.

getRowList(…)

gibt eine List<string> oder generische List<T> von der angegebenen Spalte zurück. Man erhält quasi aus dem Select eine gewünschte Spalte

getRowStringBuilder(…)

ähnlich wie die getRowList(…) ist der Rückgabewert aber ein Stringbuilder, in welchen alle Zeilen einer selektierten Spalte enthalten sind.

getDynamicList(…)

erfordert eine Klasse welche dieselben Datentypen und Bezeichnung hat wie das SQL Select. Als Rückgabe erhält man man eine List<meineKlasse>, welche 1:1 so viele Elemente und Spalten hat wie das Sql Query. Das ganze arbeitet nicht

mit Reflektionen, sondern nach dem Prinzip von diesem genialen Autor: KLICK

Dazu jeweils ein Beispiel. Ausgehend vom folgenden Select:

```
SELECT TOP 20 [ContactTypeID]
        ,[Name]
        ,[ModifiedDate]
  FROM [AdventureWorks2014].[Person].[ContactType]
```

100 %

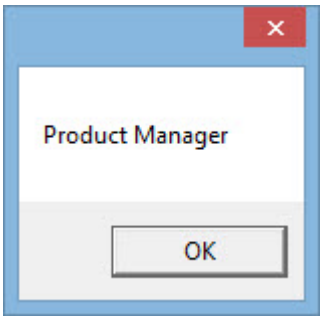
Ergebnisse | Meldungen

	ContactTypeID	Name	ModifiedDate
1	1	Accounting Manager	2008-04-30 00:00:00.000
2	2	Assistant Sales Agent	2008-04-30 00:00:00.000
3	3	Assistant Sales Representative	2008-04-30 00:00:00.000
4	4	Coordinator Foreign Markets	2008-04-30 00:00:00.000
5	5	Export Administrator	2008-04-30 00:00:00.000
6	6	International Marketing Manager	2008-04-30 00:00:00.000
7	7	Marketing Assistant	2008-04-30 00:00:00.000
8	8	Marketing Manager	2008-04-30 00:00:00.000
9	9	Marketing Representative	2008-04-30 00:00:00.000
10	10	Order Administrator	2008-04-30 00:00:00.000
11	11	Owner	2008-04-30 00:00:00.000
12	12	Owner/Marketing Assistant	2008-04-30 00:00:00.000
13	13	Product Manager	2008-04-30 00:00:00.000
14	14	Purchasing Agent	2008-04-30 00:00:00.000
15	15	Purchasing Manager	2008-04-30 00:00:00.000
16	16	Regional Account Representative	2008-04-30 00:00:00.000
17	17	Sales Agent	2008-04-30 00:00:00.000
18	18	Sales Associate	2008-04-30 00:00:00.000
19	19	Sales Manager	2008-04-30 00:00:00.000
20	20	Sales Representative	2008-04-30 00:00:00.000

einem erstellten Objekt der Klasse db:
[crayon-66299e4784e8f230658544/]
und ein string mit folgendem select:
[crayon-66299e4784e94453223409/]

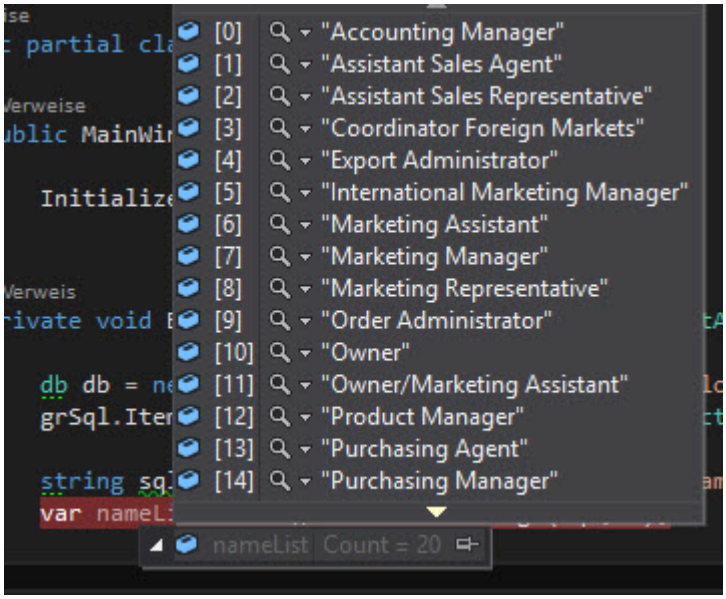
LookUp(…)

[crayon-66299e4784e96050182851/]

Antwort: 

getRowList(…)

[crayon-66299e4784e98723625752/]

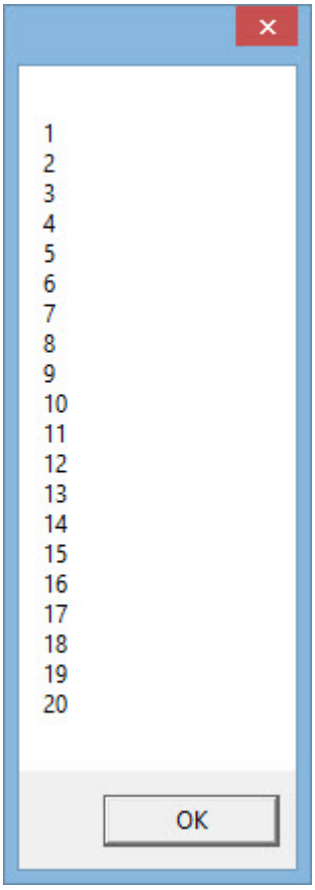


Antwort:

getRowStringBuilder(…)

[crayon-66299e4784e9a175134002/]

Antwort:

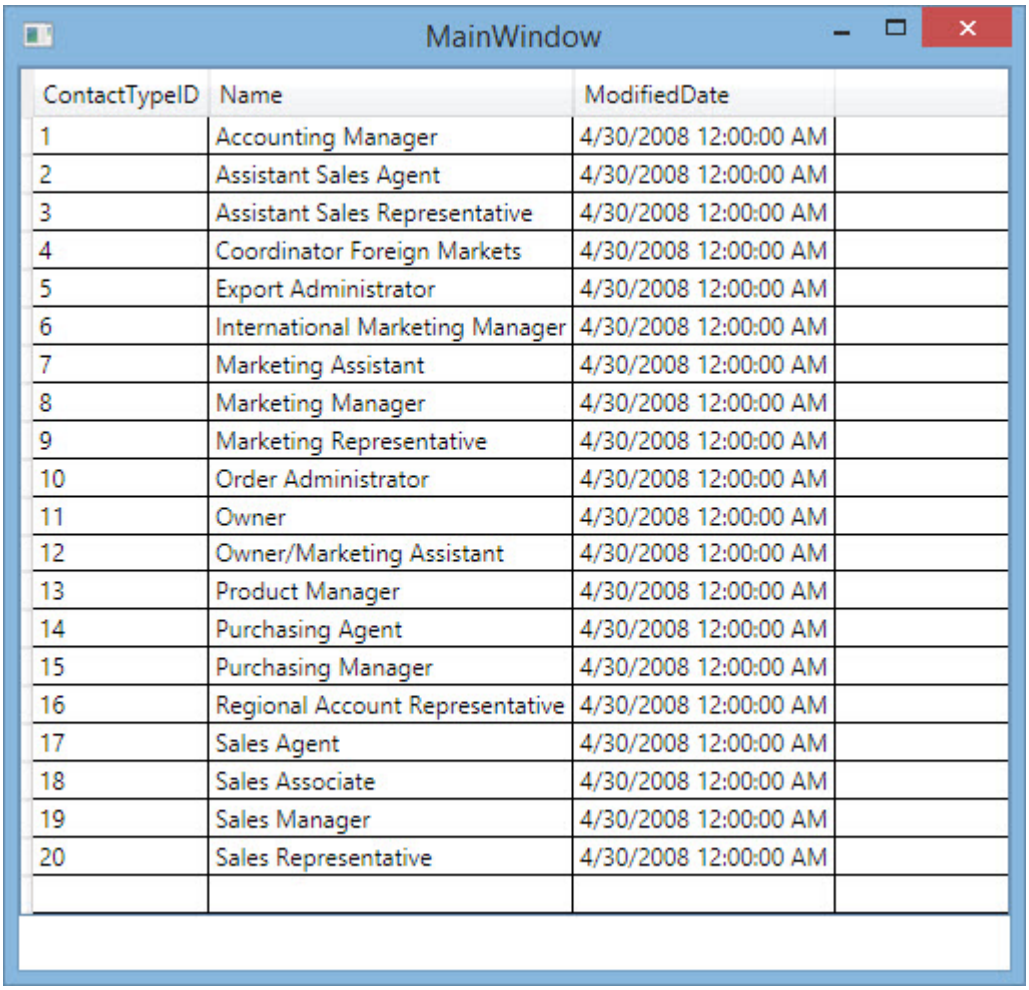


getDynamicList(…)

Wie oben bereits erwähnt, ist hierfür eine Klasse mit

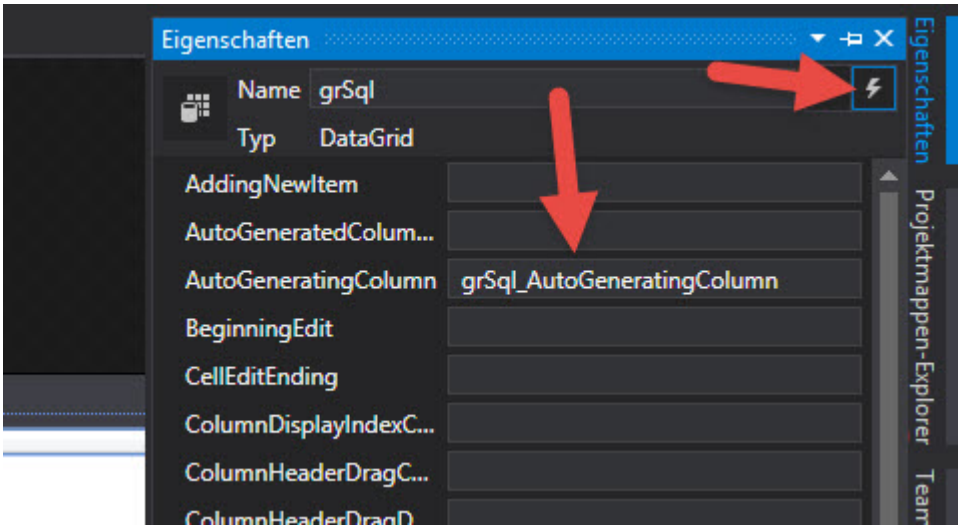
Propertys notwendig. Diese kann man ganz einfach auch mit den Methoden aus CreateClass – Klasse erstellen. Dazu weiter unten.
[crayon-66299e4784e9c429758441/]
dann kann man so eine dynamische Liste ganz einfach erstellen:
[crayon-66299e4784e9d158375847/]

diese Liste kann man nun z.B. einem Datagrid aus WPF zuordnen:
[crayon-66299e4784e9e397358802/]
[crayon-66299e4784e9f290745378/]
Das ganze sieht dann so aus:



ContactTypeID	Name	ModifiedDate	
1	Accounting Manager	4/30/2008 12:00:00 AM	
2	Assistant Sales Agent	4/30/2008 12:00:00 AM	
3	Assistant Sales Representative	4/30/2008 12:00:00 AM	
4	Coordinator Foreign Markets	4/30/2008 12:00:00 AM	
5	Export Administrator	4/30/2008 12:00:00 AM	
6	International Marketing Manager	4/30/2008 12:00:00 AM	
7	Marketing Assistant	4/30/2008 12:00:00 AM	
8	Marketing Manager	4/30/2008 12:00:00 AM	
9	Marketing Representative	4/30/2008 12:00:00 AM	
10	Order Administrator	4/30/2008 12:00:00 AM	
11	Owner	4/30/2008 12:00:00 AM	
12	Owner/Marketing Assistant	4/30/2008 12:00:00 AM	
13	Product Manager	4/30/2008 12:00:00 AM	
14	Purchasing Agent	4/30/2008 12:00:00 AM	
15	Purchasing Manager	4/30/2008 12:00:00 AM	
16	Regional Account Representative	4/30/2008 12:00:00 AM	
17	Sales Agent	4/30/2008 12:00:00 AM	
18	Sales Associate	4/30/2008 12:00:00 AM	
19	Sales Manager	4/30/2008 12:00:00 AM	
20	Sales Representative	4/30/2008 12:00:00 AM	

wem das Datumsformat stört, der kann dem Ereignis AutogeneratingColumn aus dem Datagrid eine Änderung des Datumsformates durchführen:



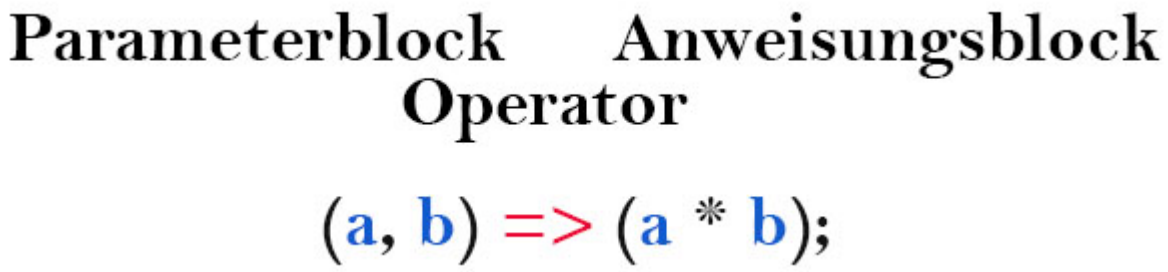
folgendes soll nun passieren, wenn das Ereignis eintrifft:
[crayon-66299e4784ea0973097518/]
Nun sieht das ganze so aus:

ContactTypeID	Name	ModifiedDate
1	Accounting Manager	30.04.2008
2	Assistant Sales Agent	30.04.2008
3	Assistant Sales Representative	30.04.2008
4	Coordinator Foreign Markets	30.04.2008
5	Export Administrator	30.04.2008
6	International Marketing Manager	30.04.2008
7	Marketing Assistant	30.04.2008
8	Marketing Manager	30.04.2008
9	Marketing Representative	30.04.2008
10	Order Administrator	30.04.2008
11	Owner	30.04.2008
12	Owner/Marketing Assistant	30.04.2008
13	Product Manager	30.04.2008
14	Purchasing Agent	30.04.2008
15	Purchasing Manager	30.04.2008
16	Regional Account Representative	30.04.2008
17	Sales Agent	30.04.2008
18	Sales Associate	30.04.2008
19	Sales Manager	30.04.2008
20	Sales Representative	30.04.2008

Lambda Ausdrücke oder anonyme Methoden

Lambda Ausdrücke werden in Kombination mit Delegaten dazu verwendet um schnelle Anweisungen auszuführen und sich dabei jede Menge Code zu ersparen.

Ohne Lambda und anonymer Methode für ein Delegatenaufruf in etwa so aussehen:



Neue Eigenschaft vom Typ delegate erstellen. (Rückgabewert ist ein int, mit 2 in Parameter a und b)
[crayon-66299e478539b887349308/]
Dann müssen wir eine Methode haben, die ebenfalls denselben Rückgabewert und Parameter besitzt (Anzahl und Typ sind entscheidend)
[crayon-66299e478539f017932212/]
Nun könnte man ein Objekt vom den oben erstellten MyDelegate erstellen und dem die Methode addiere zuweisen:
[crayon-66299e47853a0529530427/]
Durch den Aufruf des Objektes dlg zeigt der Delegat auf die Methode addiere, führt diese aus und gibt den Rückgabewert zurück:
[crayon-66299e47853a1748904845/]

Dies ist in der Tat etwas umständlich. Vor allem benötigt man immer ein Methode, die man dem Delegaten immer zuweisen muss.

schneller geht es mit den Lambda Ausdrücken.:
[crayon-66299e47853a2990709934/]
Die delegaten delegat1 bis delegat 5 tun immer dasselbe, nur wird der Code dadurch sehr kompakter und wenn man delegaten erstmal verstanden hat, wird es auch übersichtlicher.

Hat man einen Delegaten ohne Rückgabewert, würde der Lambdaausdruck mit Klammer auf- Klammer zu sein:
[crayon-66299e47853a4913734773/]

Wegen den => Operator werden die Lampdaausdrücke oft mit Linq verwechselt. Das eine hat mit dem anderen aber nichts zu tun

---

XAML Assembly Version an window Title binden

Um die Assembly Version irgendwo im Programm optisch darzustellen, kann man diese z.B. im Titelbereich des windows anzeigen lassen.

1. in app.xaml.cs folgende Eigenschaft hinzufügen:
[crayon-66299e47855ee500784704/]
2. Der XAML Code zum binden sieht dann so aus
[crayon-66299e47855f2978920438/]