C# Application Development

Database Programming with C#

Jyväskylä University of Applied Sciences, School of IT, 2008

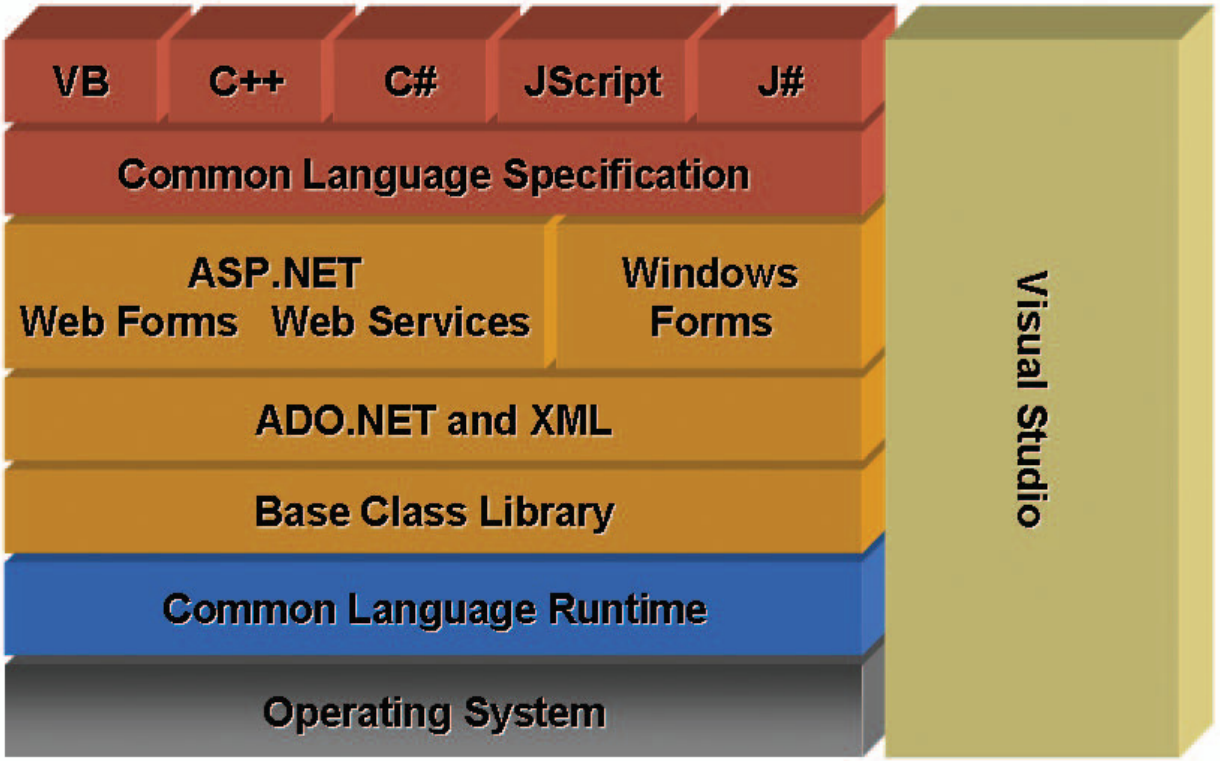
Michal Zábovský

Department of Informatics
Faculty of Management Science and Informatics
University of Zilina Slovak Republic

Overview

- Basic ADO.NET facts
- Accessing data with ADO.NET
- Using Command and DataReader objects
- Using DataAdapter and DataSet objects
- Databinding
- Examples

.NET Framework



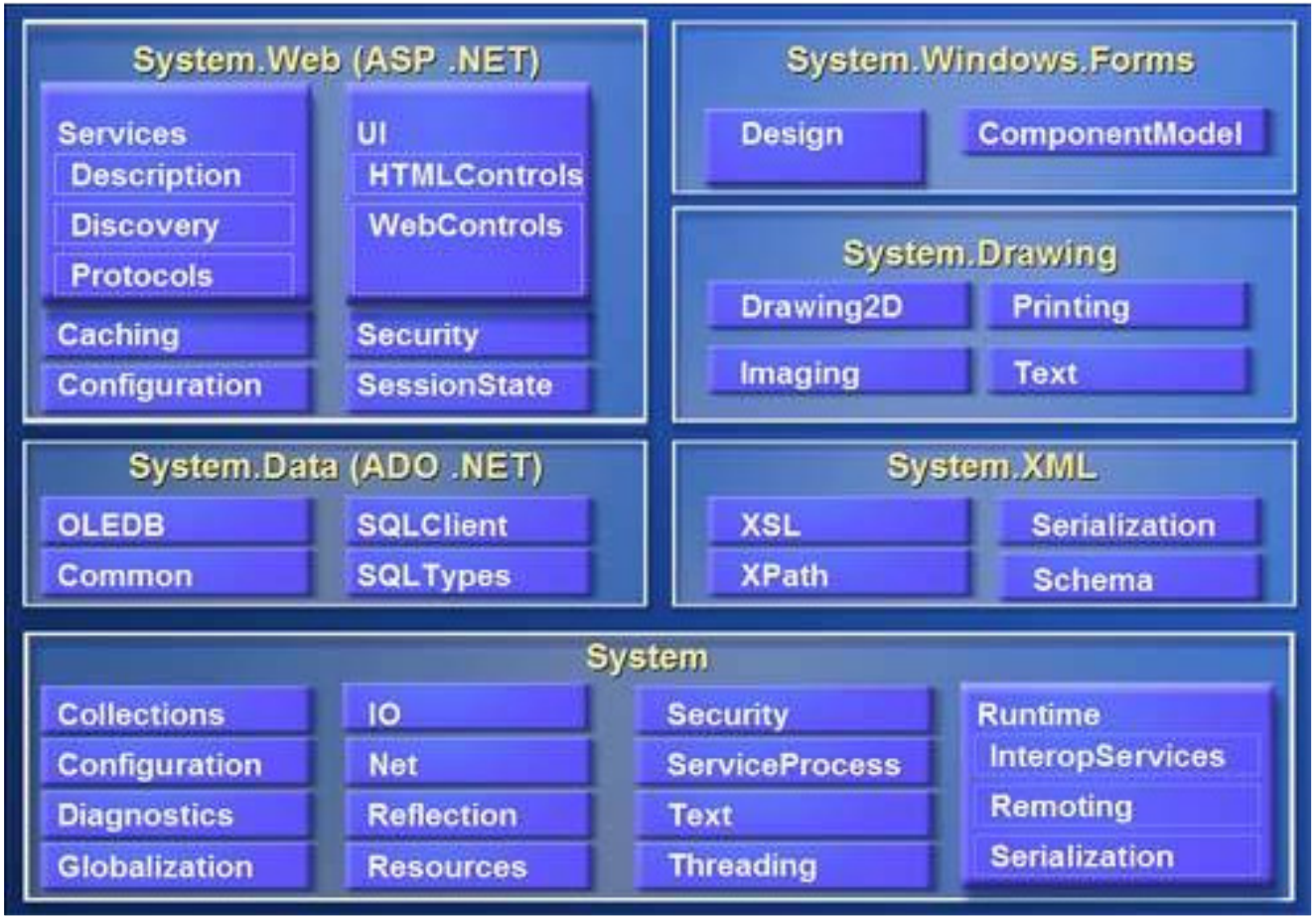
.NET versions

Windows Server
Codename "Longhorn"

	Fx 1.0	Fx 1.1	Fx 2.0	Fx 3.0	Fx 3.5
Engine Core	CLR 1.0	CLR 1.1	CLR 2.0		
	VB .NET, C#...	VB .NET, C#, C++, J#...	VB .NET, C#, C++, J#... Generics...		LINQ, C#, ...
Functional Libraries	ADO .NET 1.0 ASP .NET 1.0 XML ASMX ...	ADO .NET 1.1 ASP .NET 1.1 XML ASMX ... WSE 2.0	ADO .NET 2.0 ASP .NET 2.0 XML ASMX ... WSE 3.0	WCF WPF WF WCS	ASP.NET AJAX
Platform Technology	Windows 95 to Windows XP	+ Windows Server 2003	+ SQL 2005	Win XP SP2, WinSrv 2k3 SP1, Vista	Windows XP Win Srv 2k3, Vista, Longhorn
Developer tools (Visual Studio)	.NET 2002	.NET 2003	2005	2005 + Fidalgo + Cypress + Atlas	Orcas

2007

.NET Framework Class Library



CTS internal datatypes

Datatype	VB.NET	C#	Managed Extensions for C++
System.Byte	Byte	byte	unsigned char
System.SByte	SByte	sbyte	signed char
System.Int16	Short	short	short
System.Int32	Integer	int	int / long
System.Int64	Long	long	__int64
System.UInt16	UShort	ushort	unsigned short
System.UInt32	UInteger	uint	unsigned int / unsigned long
System.UInt64	ULong	ulong	unsigned __int64
System.Single	Single	float	Float
System.Double	Double	double	Double
System.Object	Object	object	Object^
System.Char	Char	char	wchar_t
System.String	String	string	String^
System.Decimal	Decimal	decimal	Decimal
System.Boolean	Boolean	bool	Bool

ADO.NET what says Microsoft?

ADO.NET provides consistent access to data sources, such as Microsoft SQL Server, as well as data sources exposed through **OLE DB** and **XML**. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, manipulate, and update data.

ADO.NET cleanly factors data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly or placed in an ADO.NET DataSet object in order to be exposed to the user in an ad-hoc manner, combined with data from multiple sources, or remoted between tiers. The ADO.NET DataSet object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML.

Source: Microsoft Visual C# 2005 Express Edition – Build a Program Now!

Accessing data with ADO.NET

Most of the applications must use some sort of data store. ActiveX Data Objects .NET (ADO.NET) is the technology used in the .NET Framework for database access. ADO.NET is the set of COM components (DLLs) that allows to access databases, emails or filesystem.

Before .NET

- ActiveX Data Objects (ADO) – designed for disconnected environment
- Open Database Connectivity (ODBC)
- Native drivers

Note: There is still quite confusing behavior of Microsoft in the field of technology naming. Historically, you can meet different technologies for names e.g. ActiveX or COM.

ADO and ADO.NET objects

ADO

- Connection - set properties and call *Open* method to connect database
- Command - create object that holds SQL statement
- RecordSet

ADO.NET

- Connection – set properties and call *Open* method to connect database
- Command – create object that holds SQL statement, supports parameters
- DataReader – for read-only, forward-only access (ForwardOnly cursor in ADO)
- DataAdapter –object between database and DataSet, is responsible for keeping track of the original data since you last connected
- DataSet – in-memory representation of data, it doesn't directly connect to a database

A few more objects than ADO. Additional objects improve flexibility in application design.

Access types and data groups

- Direct database access
- Connectionless data access
  - Information is stored into computer's (client's) memory
  - Useful for architectures using mobile devices (e.g. PDAs)

For both access types are defined two class families:

- Smart data – typically implemented with business objects
- Raw data – data from database are locally stored (XML is used to marshal data to and from ADO.NET)

`System.Data` – namespace has all the classes you need to access database or data store

Core ADO.NET namespaces

- `System.Data.SqlClient` – optimized for data access with SQLServer
- `System.Data.OleDb` – optimized for OLE DB (Object Linking and Embedding for Databases) data access to databases other than SQLServer (MS Access, Excel, dBase)
- `System.Data.Odbc` – to connect to ODBC data sources using an ODBC connection. Is better to use OLE DB if it's presented for particular database (in the .NET environment is ODBC a bit slower than OLE DB).

Simple example – What we need?(1)

- Northwind database installed on SQL Server
  - Download example database from Microsoft web site
    http://www.microsoft.com/downloads/details.aspx?familyid=06616212-0356-46a0-8da2-eebc53a68034&displaylang=en
  - Execute file **SQL2000SampleDb.msi** to install data files
  - In Microsoft SQL Server Management Studio
    - In the **Object Explorer** tree right click **Databases->Attach**
    - Click **Add** button and choose database file for Northwind database (default path is **c:\SQL Server 2000 Sample Databases\NORTHWND.MDF**)
    - Click **OK** button

Simple example – What we need?(2)

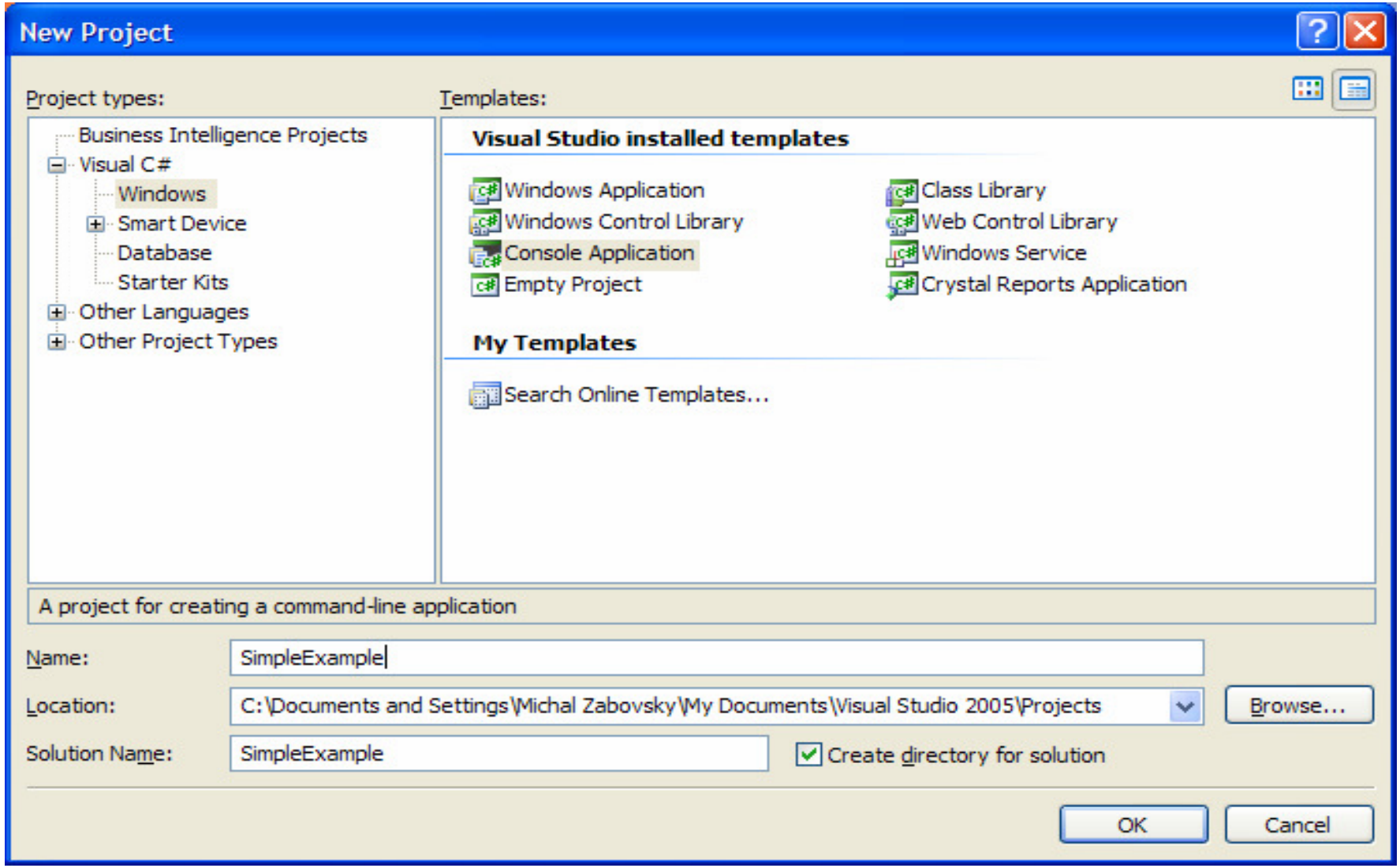
User account that allows connection to Northwind database

- In **Object Explorer** tree click **Security** then right click **Logins** and choose **New Login**
- Put **Login name**, choose **Server authentication** and type **Password** uncheck **User must change password at next login** option
- Set **Default database** to Northwind
- In the **User Mapping** page check Northwind map and check following **Database role memberships** (public role is already checked):
  - db_datareader
  - db_datawriter
  - db_ddladmin
- Click **OK** button

Simple example – What we need?(3)

- Microsoft Visual Studio 2005 C#
  - Start MS Visual Studio 2005
  - Choose **File->New->Project** item from main menu
  - Choose **Console Application** from C# project type and name it **SimpleExample**
  - Click **OK** button

New C# project



Writing application code (1)

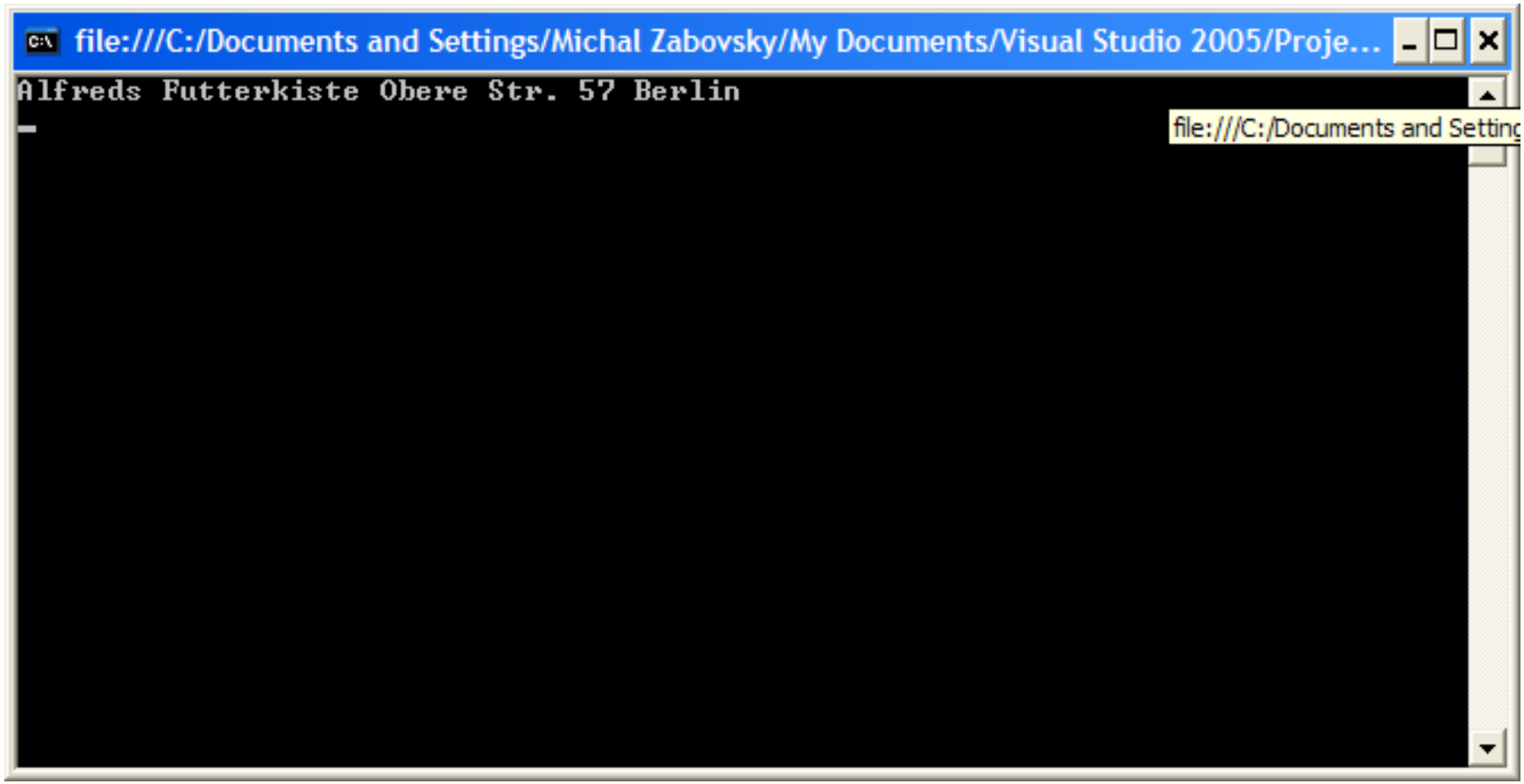
```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data.SqlClient;

namespace SimpleExample
{
    class Program
    {
        static void Main(string[] args)
        {
            try {
                SqlConnection connection = new SqlConnection (
                    "server=localhost; database=Northwind; " +
                    "uid=coder; pwd=access"
                    );
                SqlCommand command = connection.CreateCommand ();
                command.CommandText =
                    "SELECT CompanyName, Address, City " +
                    "FROM Customers " +
                    "WHERE CustomerID = 'ALFKI'";
```

Writing application code (2)

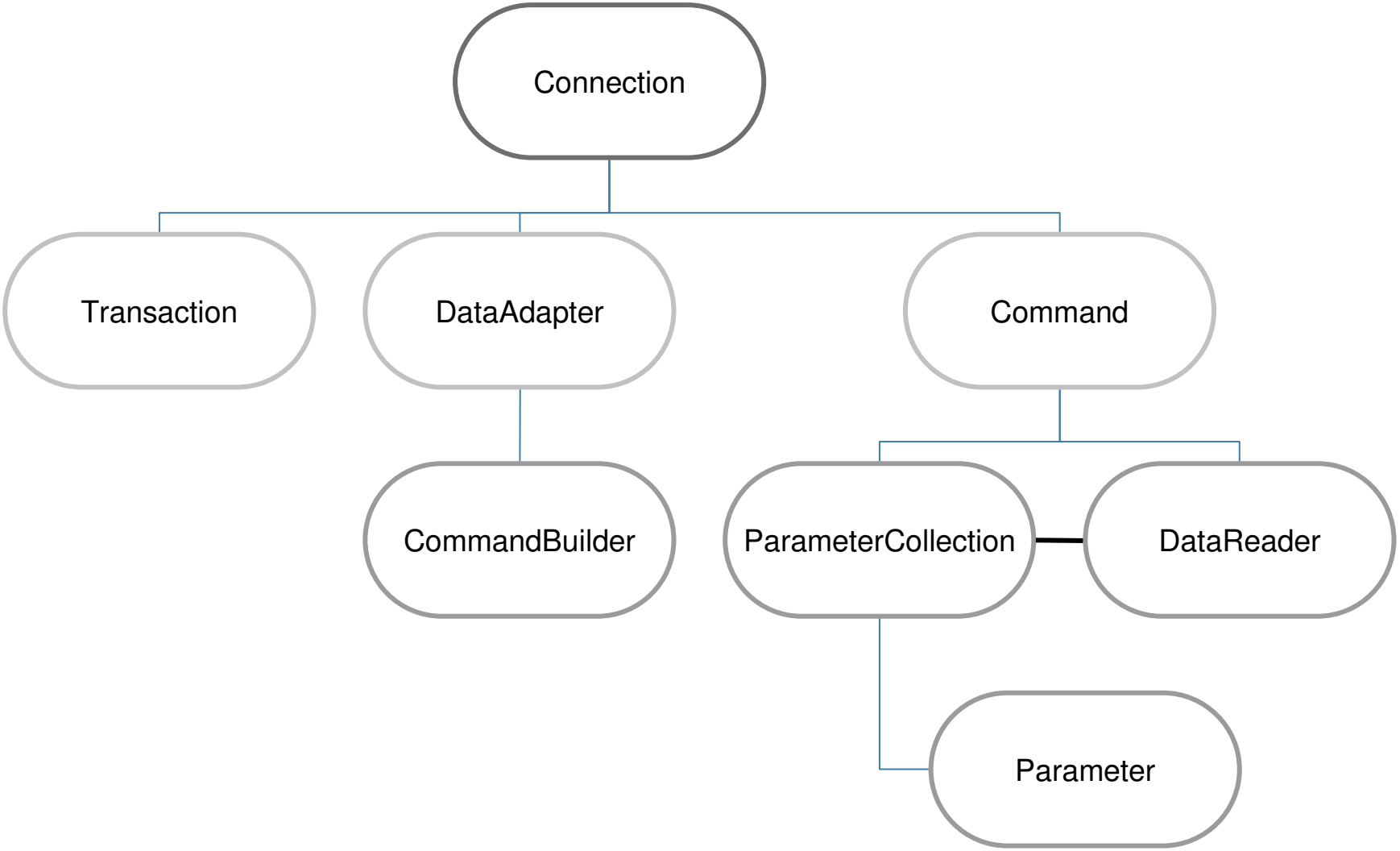
```
                connection.Open ();
                SqlDataReader dataReader = command.ExecuteReader ();
                dataReader.Read ();

                Console.WriteLine (dataReader["CompanyName"] + " " +
            dataReader["Address"] + " " + dataReader["City"]);

                dataReader.Close ();
                connection.Close ();
            } catch (SqlException e) {
                Console.WriteLine ("Exception: " + e.Message);
            }
            Console.ReadLine ();
        }
    }
}
```

Now run application by choosing menu item **Debug->Start debuging** or by pressing **F5** button.

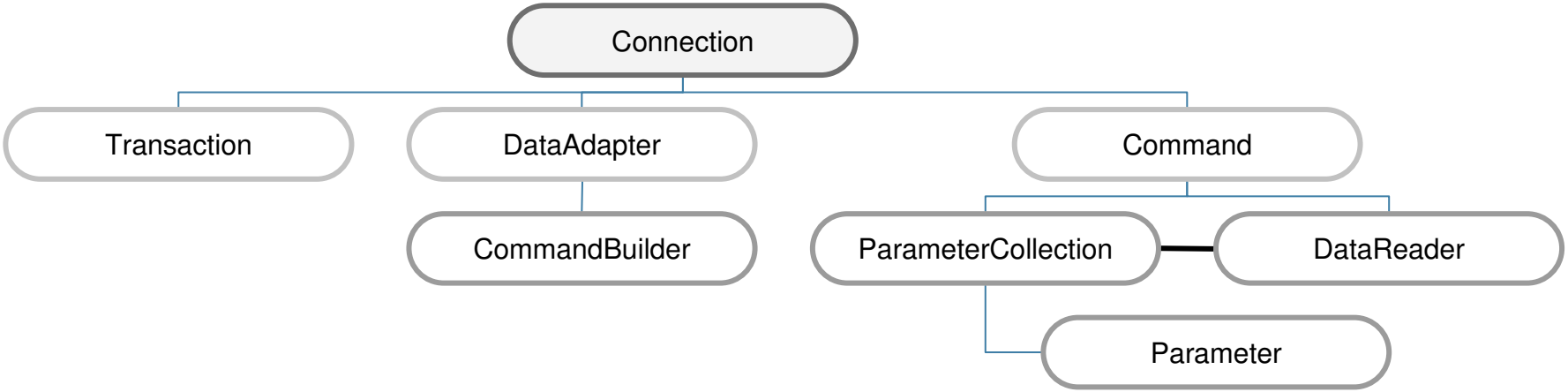
Simple Example



Data access classes



Connection class



To work with any database, the first thing you must to do is to connect to it. In ADO.NET, you can use the **Connection** object for this. There are three basic types of **Connection** object:

- SqlConnection
- OleDbConnection
- OdbcConnection

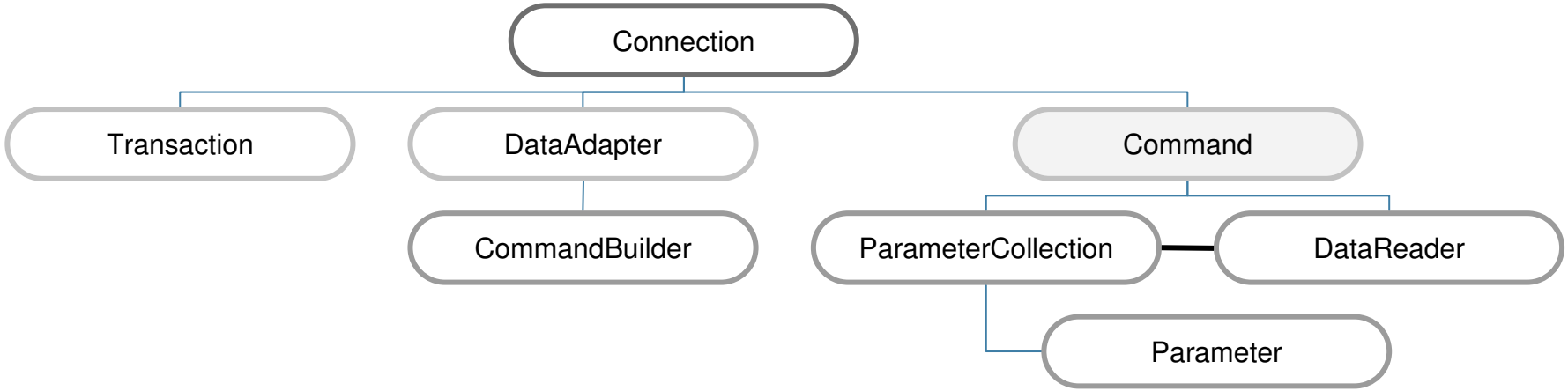
When you open a **Connection** object, you must always explicitly close it. Calling **Close** or **Dispose** on a **Connection** object ensures you that the connection is sent back to the connection pool.

SqlConnection class

Basic properties of **SqlConnection** object are:

- **ConnectionTimeout** – timeout for the connection
- **Database** - the name of the current database
- **DataSource** – the name of SQL Server instance to which you are going to connect
- **ServerVersion** – the version of SQL Server instance
- **State** – current state of the connection
- **WorkstationId** – database client ID

Command class



The **Command** object is used to execute SQL statements against a database. The SQL statements can be ad hoc text or the name of a stored procedure in SQL Server.

- SqlCommand
- OleDbCommand
- OdbcCommand

The **Command** object can be created i two ways – by calling the **CreateCommand** method of a **Connection** object or by creating of an instance of the **SqlCommand** or **OleDbCommand** and by passing a valid **Connenction** object to the **Command** instance.

SqlCommand properties

Basic **SqlCommand** properties:

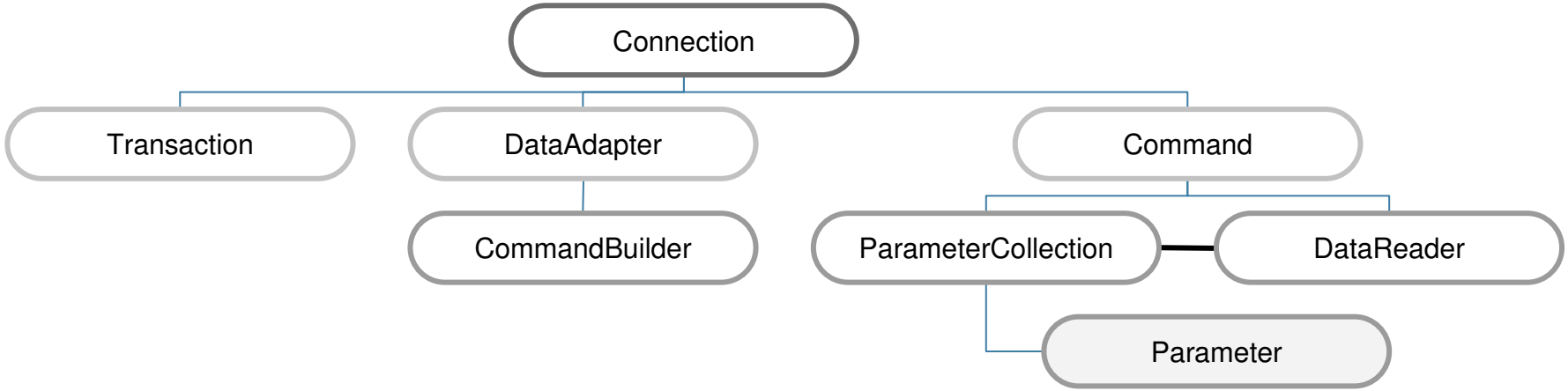
- **CommandText** – the SQL statement or stored procedure
- **CommandTimeout** – time before terminating an attempt to execute
- **CommandType** – indicates, how the CommandText property is interpreted
- **Connection** – an instance of the Command object
- **Parameters** –SqlParameterConnection object collection
- **Transaction** – the transaction in which the SqlCommand is executed
- **UpdateRowSource** – indicates, how command results are applied to the DataRow when Update method of DataAdapter is used

SqlCommand execute methods

Execute methods of an **SqlCommand** object:

- **ExecuteReader** – to execute commands that return rows
- **ExecuteNonQuery** – to execute commands such as INSERT, DELETE, UPDATE or SET
- **ExecuteScalar** – method retrieves a single value from database
- **ExecuteXmlReader** – method is used to build an XmlReader object

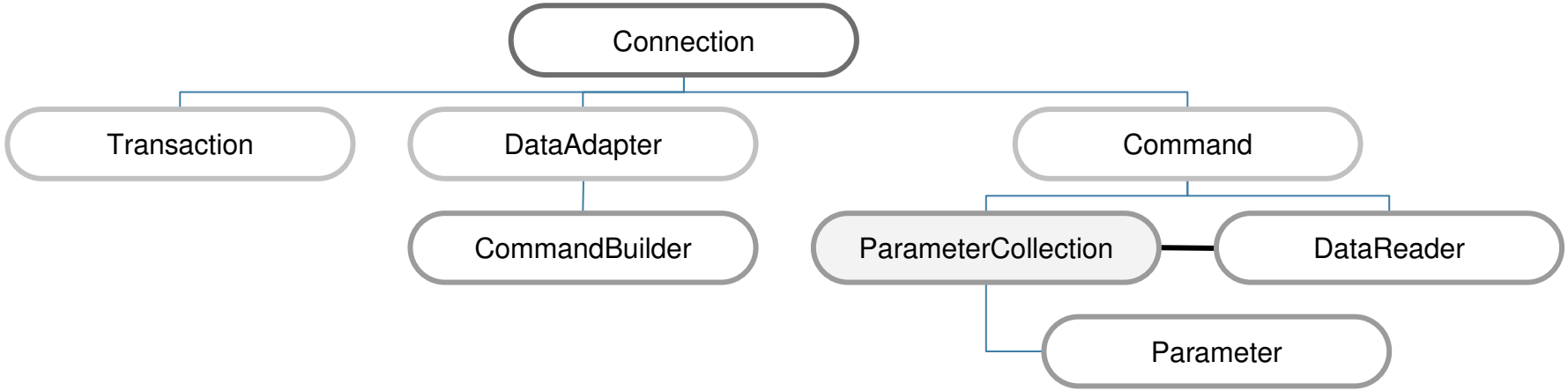
Parameter class



Parameter object is used to passing parameter to a **Command** object. Parameter value can be passed to SQL command or to stored procedure.

- SqlParameter
- OleDbParameter
- OdbcParameter

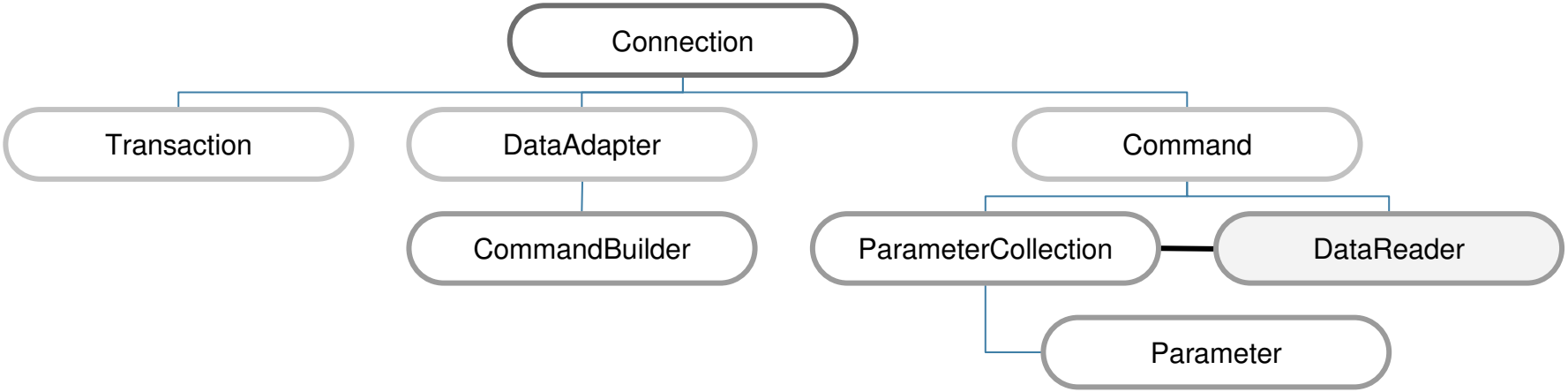
ParameterCollection class



ParameterCollection is data structure used to passing more than one **Parameter** object to a **Command** object.

- SqlParameterCollection
- OleDbParameterCollection
- OdbcParameterCollection

DataReader class



DataReader instance is used to read rows returned as the result of the **Command** object.

- SqlDataReader
- OleDbDataReader
- OdbcDataReader

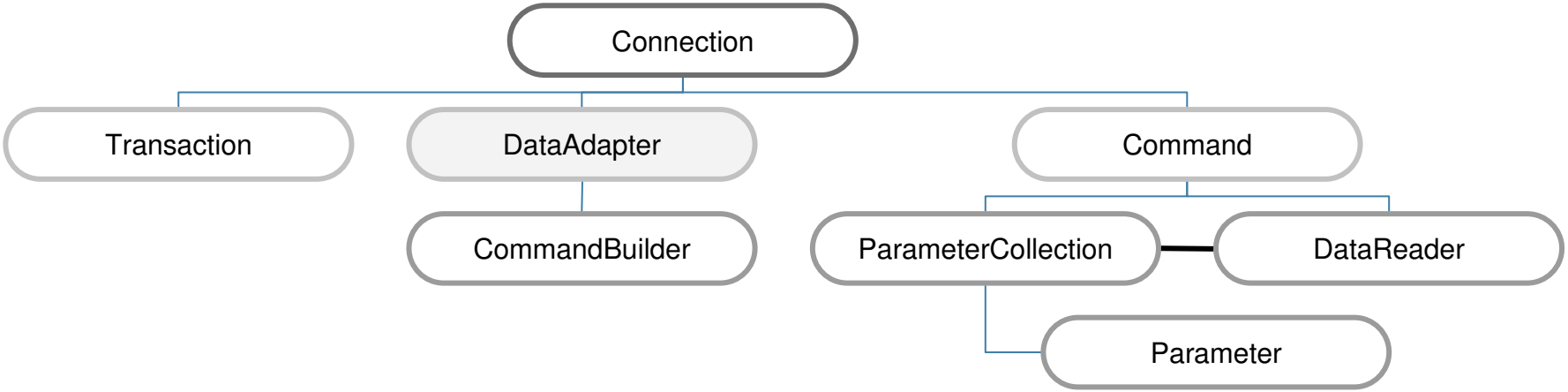
DataReader is a forward-only set of records, so you can't move backward in the **DataReader** instance. On the other hand, reading data by using **DataReader** is obviously faster than by using **DataSet**.

SqlDataReader methods

SqlDataReader methods are used to reading data into appropriate data type:

- **GetSqlBinary** - gets the value of the specified column as a *SqlBinary*
- **GetSqlBoolean** - gets the value of the specified column as a *SqlBoolean*
- **GetSqlByte** - gets the value of the specified column as a *SqlByte*
- **GetSqlDateTime** - gets the value of the specified column as a *SqlDateTime*
- **GetSqlDecimal** - gets the value of the specified column as a *SqlDecimal*
- **GetSqlDouble** - gets the value of the specified column as a *SqlDouble*
- **GetSqlGuid** - gets the value of the specified column as a *SqlGuid*
- **GetSqlInt16** - gets the value of the specified column as a *SqlInt16*
- **GetSqlInt32** - gets the value of the specified column as a *SqlInt32*
- **GetSqlInt64** - gets the value of the specified column as a *SqlInt64*
- **GetSqlMoney** - gets the value of the specified column as a *SqlMoney*
- **GetSqlSingle** - gets the value of the specified column as a *SqlSingle*
- **GetSqlString** - gets the value of the specified column as a *SqlString*

DataAdapter class



If you need more flexible features than a **DataReader** offers, you can use a **DataSet** object as a container for records from the database. Data into a **DataSet** are loaded by a **DataAdapter.** The synchronization is provided by a **Connection** object.

- SqlDataAdapter
- OleDbDataAdapter
- OdbcDataAdapter

The **DataSet** :

- Doesn't connect to a database
- Simply holds data and table information in its **DataTables** collection

DataSet example code (1)

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data;

using System.Data.SqlClient;

namespace SelectIntoDataSet {
  class Program {
        static void Main (string[] args)
        {
            DataSet dataSet = new DataSet ();

            try {
                String connectionString =
                    "server=localhost;database=Northwind;" +
                    "uid=coder;pwd=access";
                SqlConnection connection =
                    new SqlConnection (connectionString);

                String query =
                    "SELECT TOP 10 CompanyName, Address, City " +
                    "FROM Customers " +
                    "ORDER BY CompanyName";
```

DataSet example code (2)

```
        SqlCommand command = connection.CreateCommand ();
        command.CommandText = query;

        SqlDataAdapter dataAdapter = new SqlDataAdapter ();
        dataAdapter.SelectCommand = command;

        connection.Open ();
        dataAdapter.Fill (dataSet, "Customers");
        connection.Close ();

    } catch (SqlException e) {
        Console.WriteLine ("Exception: " + e.Message);
    }
```

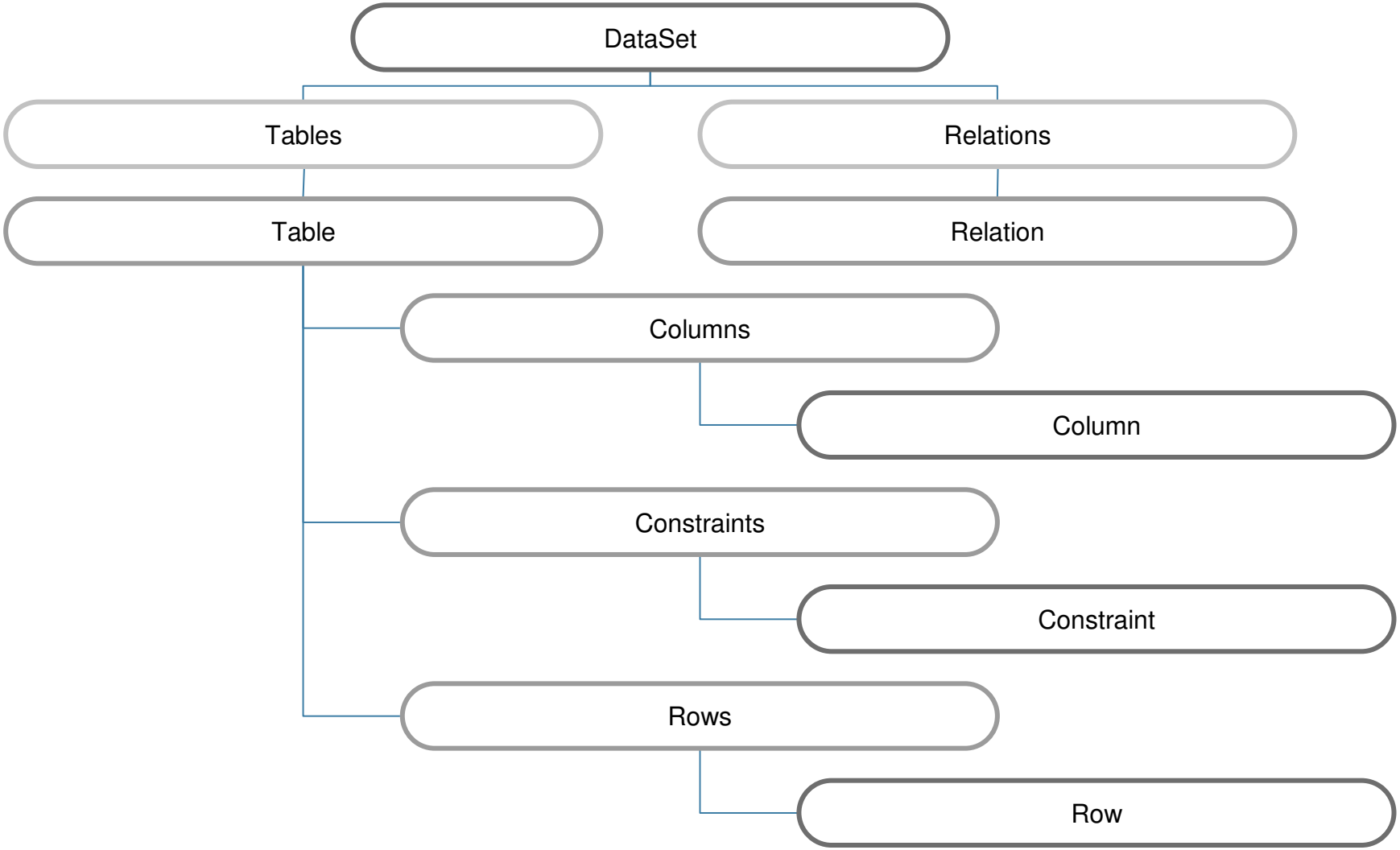
DataSet example code (3)

```
                    // Database is already disconnected
                    DataTable dataTable = dataSet.Tables["Customers"];
                    foreach (DataRow dataRow in dataTable.Rows) {
                        Console.WriteLine (dataRow["CompanyName"]);
                        Console.WriteLine (dataRow["Address"] + ", " +
                            dataRow["City"]);
                        Console.WriteLine ("------------------------------");
                    }

                    Console.ReadLine ();
                }
            }
        }
```

DataSet example

```
file:///c:/Documents and Settings/Michal Zabovsky/My Documents/Visual Studio 2005/Proje...
Alfreds Futterkiste
Obere Str. 57, Berlin
--------------------------------------------------
Ana Trujillo Emparedados y helados
Avda. de la Constitución 2222, México D.F.
--------------------------------------------------
Antonio Moreno Taquería
Mataderos  2312, México D.F.
--------------------------------------------------
Around the Horn
120 Hanover Sq., London
--------------------------------------------------
Berglunds snabbköp
Berguvsvägen  8, Lulea
--------------------------------------------------
Blauer See Delikatessen
Forsterstr. 57, Mannheim
--------------------------------------------------
Blondesddsl pcre et fils
24, place Kléber, Strasbourg
--------------------------------------------------
Bólido Comidas preparadas
C/ Araquil, 67, Madrid
--------------------------------------------------
Bon app'
12, rue des Bouchers, Marseille
--------------------------------------------------
Bottom-Dollar Markets
23 Tsawassen Blvd., Tsawassen
--------------------------------------------------
```

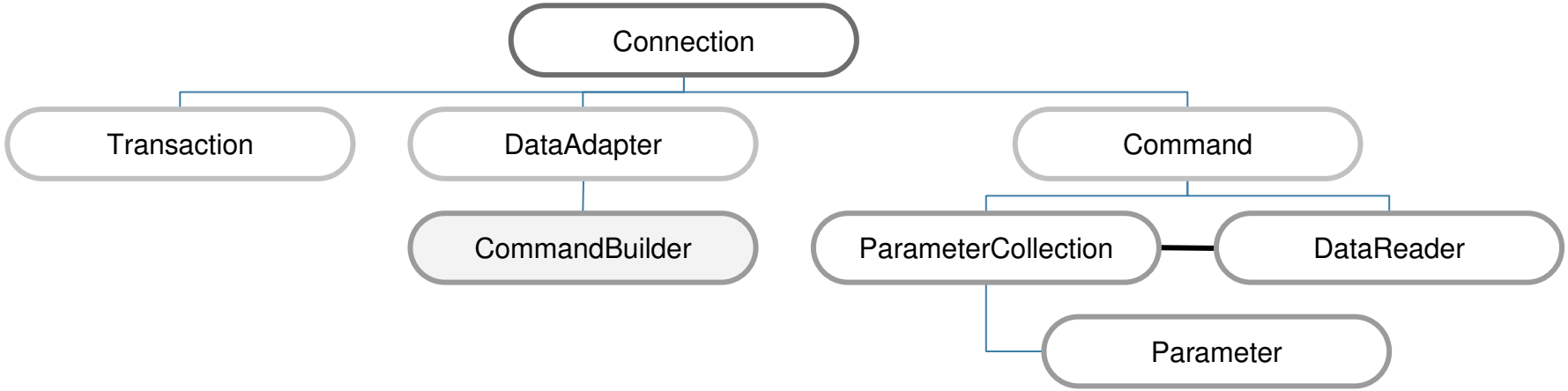
DataSet class hierarchy



Steps to fill DataSet

1. Build a connect string to database.
2. Create object **SqlConnection** and use prepared connect string with it.
3. Build a SELECT statement.
4. Create object **SqlCommand** and assign prepared SELECT statement to the **CommandText** property of this object.
5. Create object **SqlDataAdapter** and set the property **SelectedCommand** to the **SqlCommand** object.
6. Create **DataSet** object.
7. Use **Open()** method of the **SqlConnection** object to open database connection.
8. Call **Fill()** method of **SqlDataAdapter** object to reading rows from table and to save then into a **DataTable** object of the **DataSet** object.
9. Close the database connection by calling **Close()** method of **SqlConnection** object.
10. Select **DataTable** object from the **DataSet** object.
11. By using **DataRow** object show columns for each row of **DataTable** object.

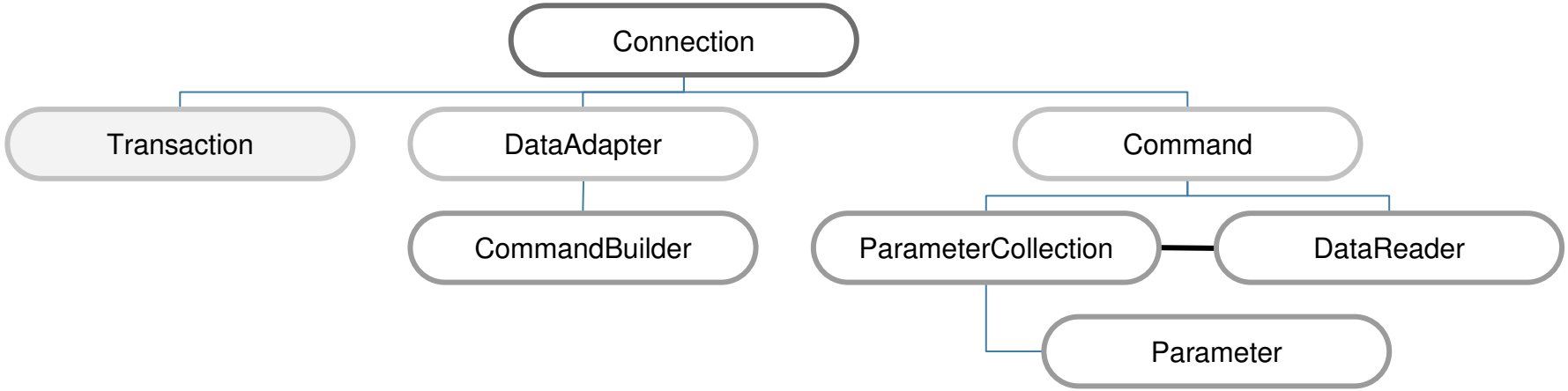
CommandBuilder class



The **CommandBuilder** object is used to create INSERT, UPDATE and DELETE commands automatically. These commands are synchronizing each change of a **DataSet** object with database. The synchronization is provided by a **DataAdapter** object.

- SqlCommandBuilder
- OleDbCommandBuilder
- OdbcCommandBuilder

Transaction class

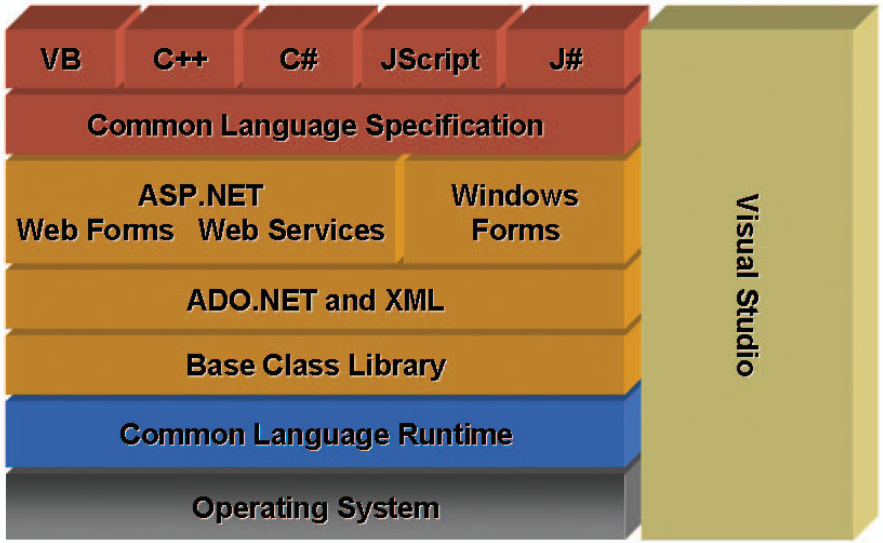


The **Transaction** object represents database transaction.

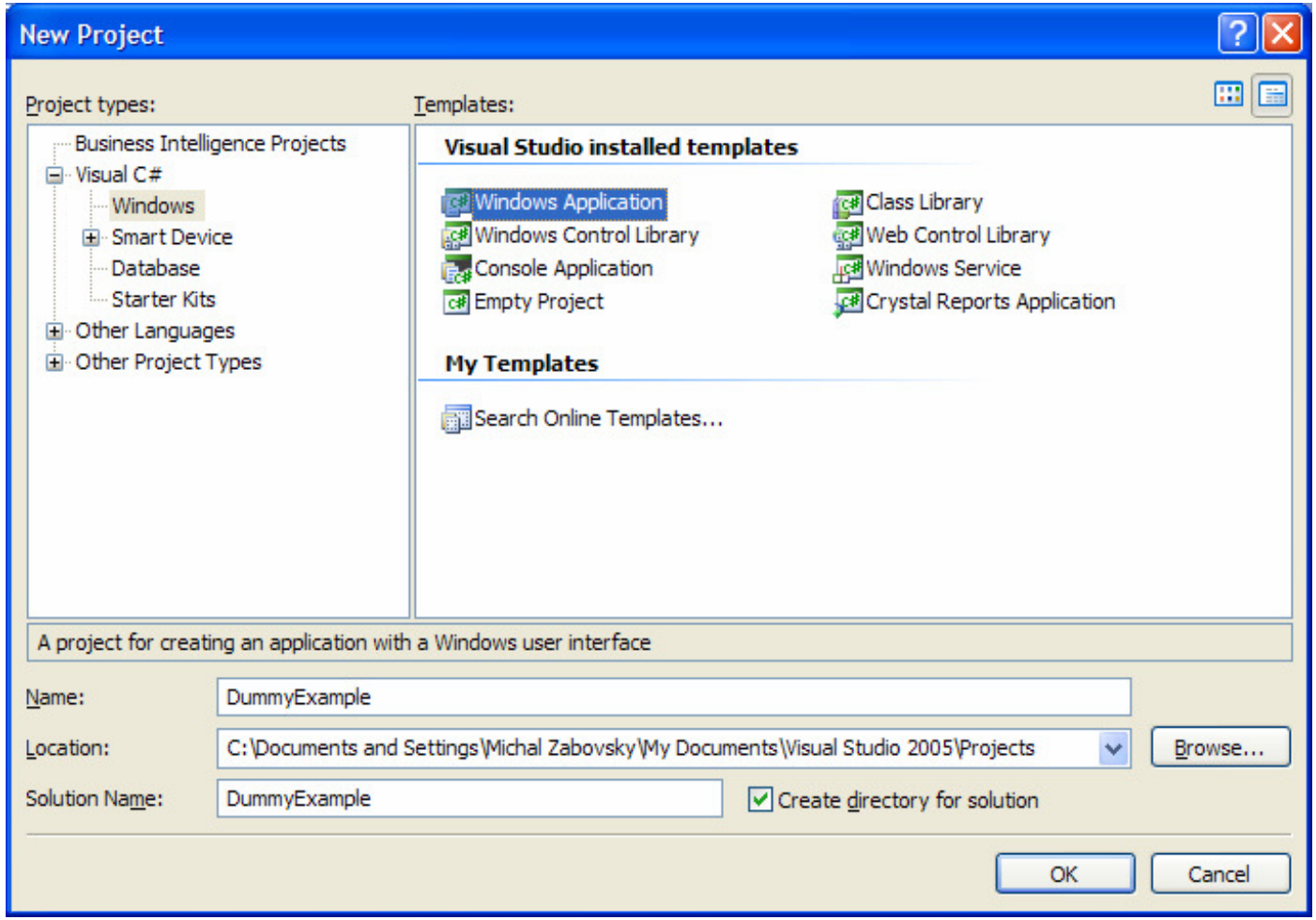
- SqlTransaction
- OleDbTransaction
- OdbcTransaction

Visual Studio .NET

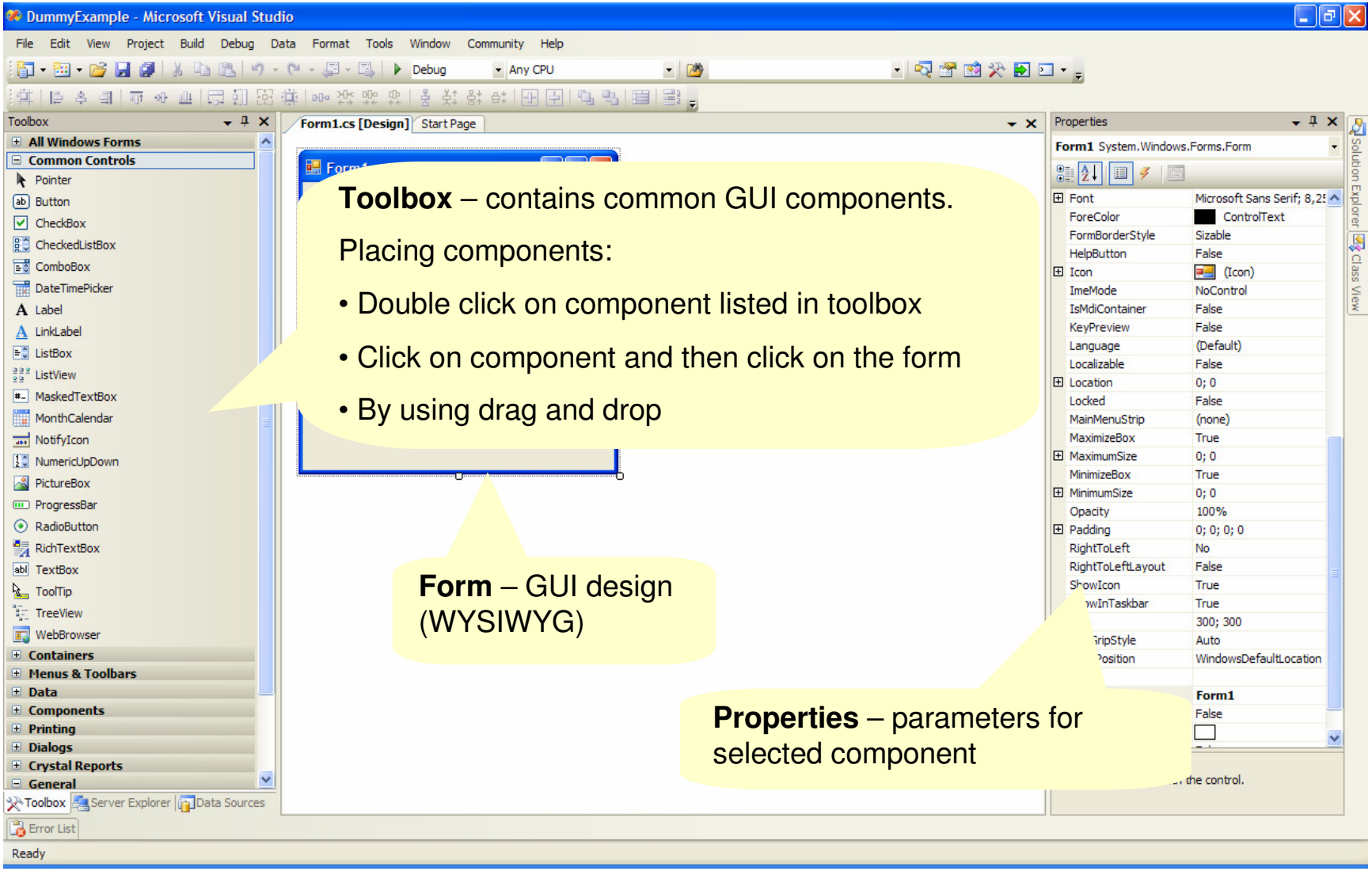
- Ability to rapidly develop and deploy form based applications
- Enterprise-level application features
- Fully objected development model



Simple GUI application



Visual Studio - Workspace



Toolbox – contains common GUI components.

Placing components:

- Double click on component listed in toolbox
- Click on component and then click on the form
- By using drag and drop

Form – GUI design (WYSIWYG)

Properties – parameters for selected component

Toolbox component groups

- **Common Controls** – basic graphical components forming user interface, e.g. Button, Label, ComboBox etc.
- **Containers** – components used for grouping other components logically, e.g. Panel, GroupBox, TabControl etc.
- **Menus & Toolbars** – menus, toolbar, statusbar
- **Data** – database related components, mostly used to present information from databases
- **Components** – other non-visual components mostly used for system operations
- **Printing** – printing components and dialogs
- **Dialogs** – common dialogs, such as OpenFileDialog etc.
- **Crystal Reports** – report-based components

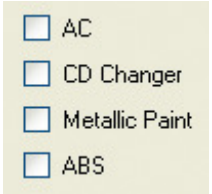
Common windows app. controls

Button – indicates that user wants to trigger an action associated with the button.

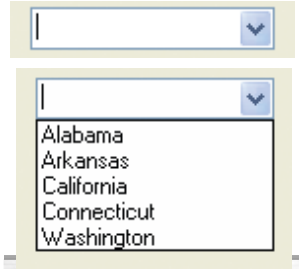
TextBox – used to get user input. Mostly used for information non restrictive in choices.

Label – simple text used to describe other controls e.g. TextBoxes.

RadioButton – offers multiple choices but user can only pick one from the list.

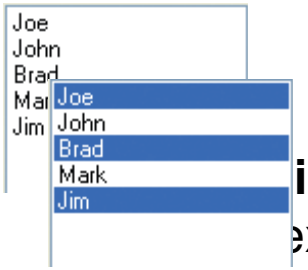


CheckBox – mostly used to identify characteristics by Boolean choice, e.g. yes/no, on/off etc.



ComboBox – drop-down list of valid choices combined with text box. Features can be modified by component properties.

Common windows app. controls

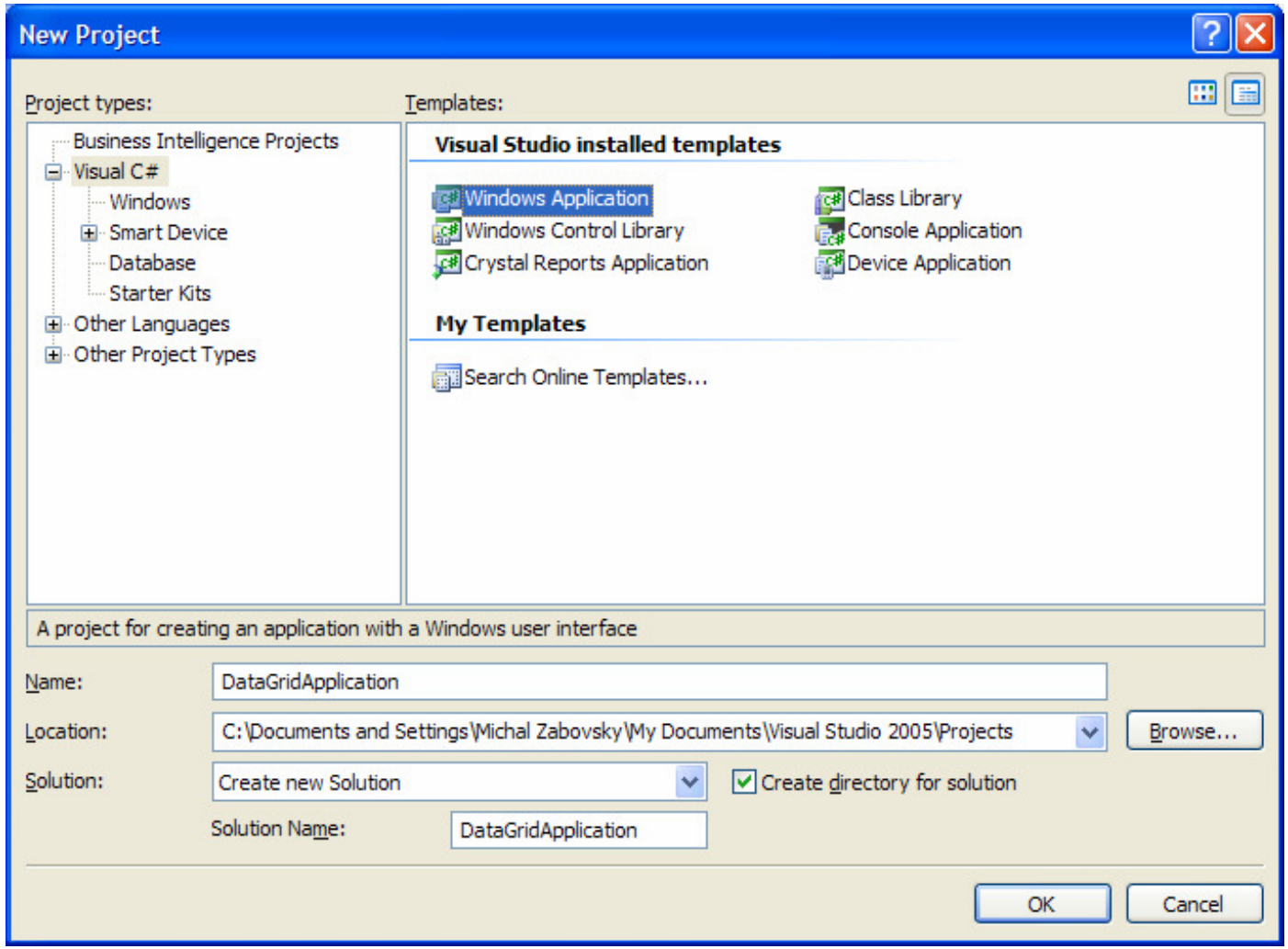


istBox – short list of valid choices. It doesn't allow the user to enter ext. The user can select more than one item when it's permitted.

ToolTip – used to display information about control when user holds mouse cursor over it.

NumericUpDown – used to select numerical values from predefined set of numbers.

DataGrid application (step 1)



DataGrid application (step 2)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Window Community Help

Debug | Any CPU

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingN
  - ReportVi
- Compon
- Printing
- Dialogs
- Crystal Reports
- General

DataGridView
Version 2.0.0.0 from Microsoft Corporation
.NET Component

Displays rows and columns of data in a grid you can customize.

Form1.cs [Design]* | Start Page

DataGrid Example

Properties

Form1 System.Windows.Forms.Form

Font	Microsoft Sans Serif; 8,25
ForeColor	ControlText
FormBorderStyle	Sizable
HelpButton	False
Icon	(Icon)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(Default)
Localizable	False
Location	0; 0
Locked	False
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0; 0
MinimizeBox	True
MinimumSize	0; 0
Opacity	100%
Padding	0; 0; 0; 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	434; 376
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	DataGrid Example
TopMost	False
TransparencyKey	

Text
The text associated with the control.

Toolbox | Server Explorer | Data Sources | Error List

Ready 15, 15 434 x 376

DataGrid application (step 3)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Window Community Help

Debug | Any CPU

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

DataGridView Tasks
- Choose Data Source: (none)
- Edit Columns...
- Add Column...
- Enable Adding
- Enable Editing
- Enable Deleting
- Enable Column Reordering
- Dock in parent container

Properties

dataGridView1 System.Windows.Forms.DataGridV

(ApplicationSettings)	
(DataBindings)	
(Name)	dataGridView1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColu	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	18
ColumnHeadersHeightSi	AutoSize
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Solution Explorer | Class View

DataGrid application (step 4)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

DataGridView Tasks

Choose Data Source (none)

None

Add Project Data Source...

Click the 'Add Project Data Source...' link to connect to data.

Properties

dataGridView1 System.Windows.Forms.DataGridV

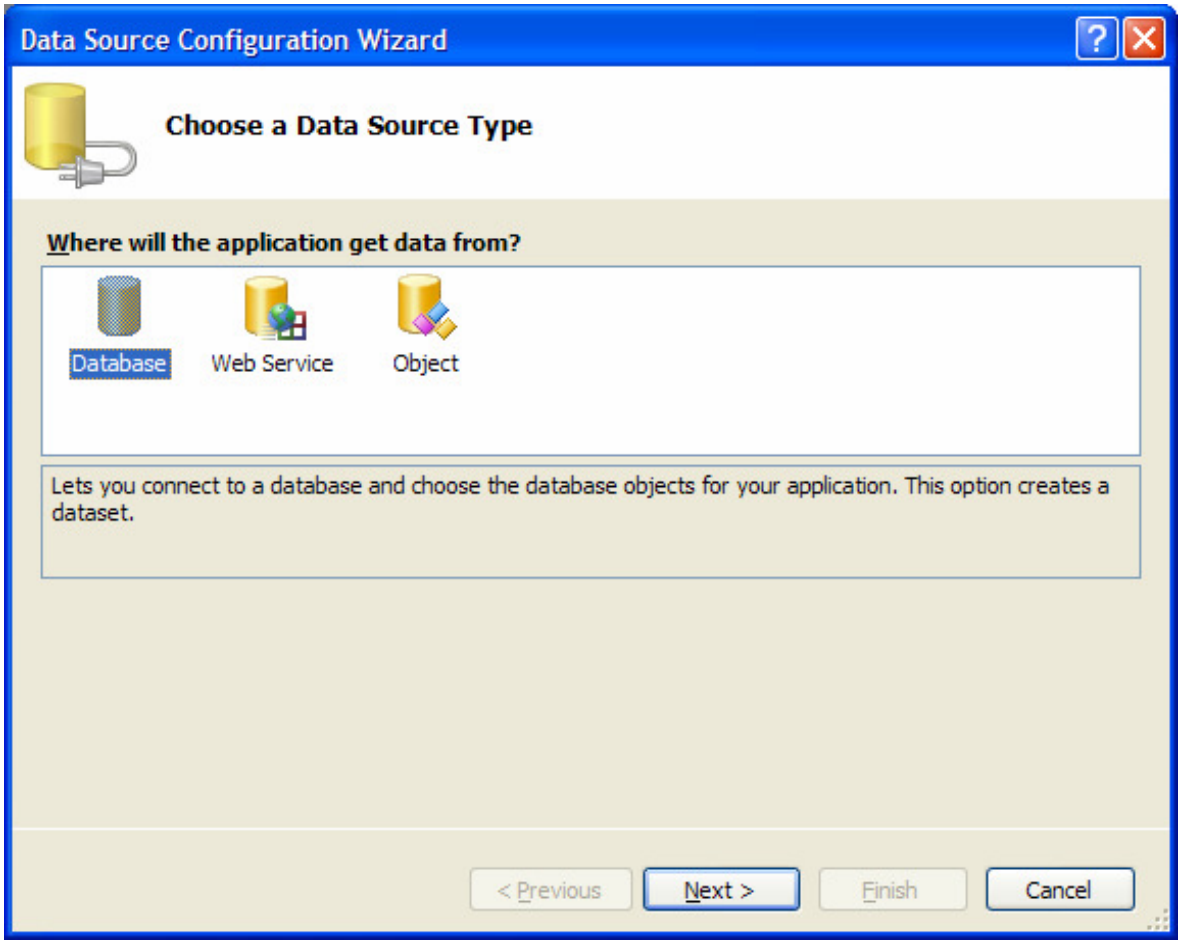
(ApplicationSettings)	
(DataBindings)	
(Name)	dataGridView1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColu	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	18
ColumnHeadersHeightSi	AutoSize
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Ready

DataGrid application (step 5)



DataGrid application (step 6)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

Data Source C

Which data c

Connection

Connection...

Cancel

Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (SqlClient) — Change...

Server name: misik — Refresh

Log on to the server

- Use Windows Authentication
- Use SQL Server Authentication

User name: coder

Password: ••••••

Save my password

Connect to a database

- Select or enter a database name:
- Logical name:

master
msdb
Northwind
tempdb

Advanced...

Test Connection | OK | Cancel

Properties

dataGridView1 System.Windows.Forms.DataGridV

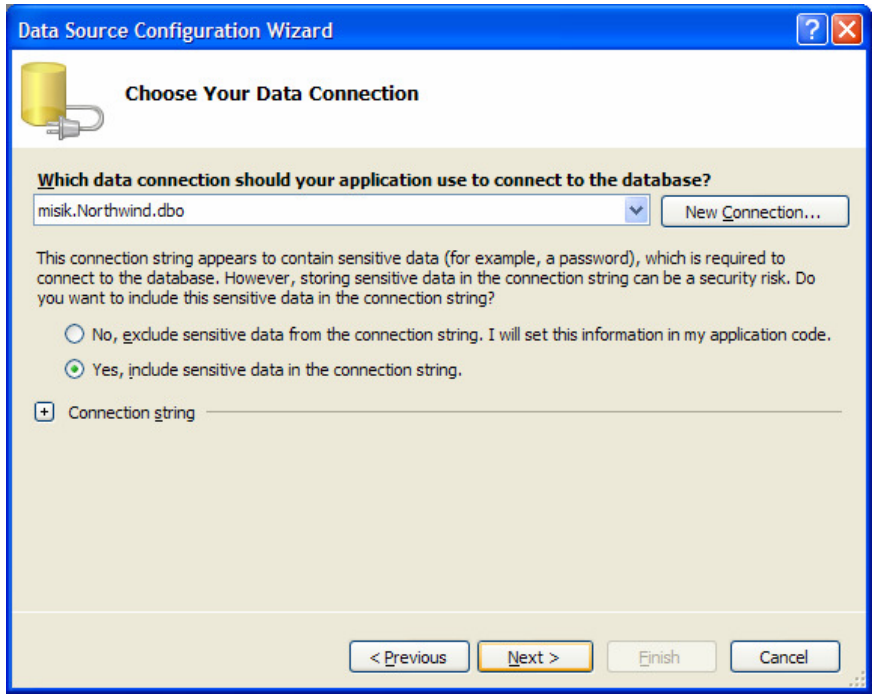
(ApplicationSettings)	
(DataBindings)	
(Name)	dataGridView1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColur	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	18
ColumnHeadersHeightSi	AutoSize
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

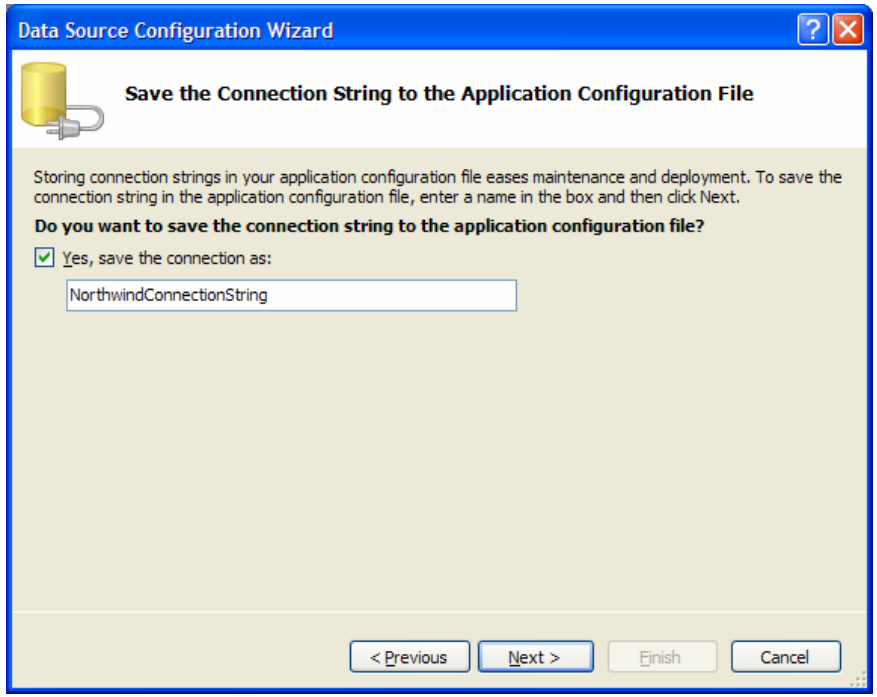
(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

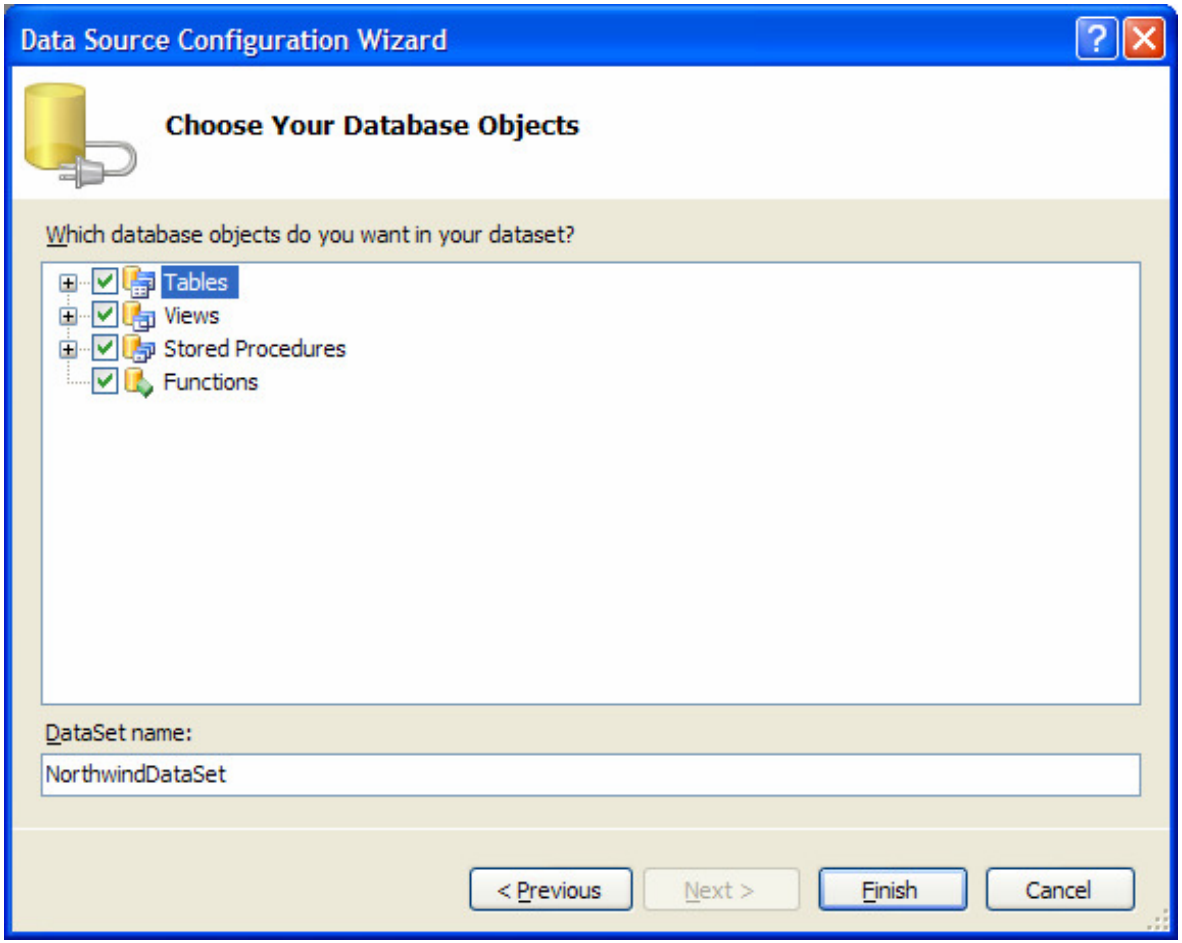
Ready

DataGrid application (step 7)





DataGrid application (step 8)



DataGrid application (step 9)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

DataGridView Tasks

Choose Data Source: (none)

- NorthwindDataSet
  - Categories
  - CustomerCustomerDemo
  - CustomerDemographics
  - Customers
  - Employees
  - EmployeeTerritories
  - Order Details
  - Orders
  - Products

Add Project Data Source...

Selecting a project data source creates an instance on the form and binds to it through a new BindingSource.

Properties

dataGridView1 System.Windows.Forms.DataGridV

(ApplicationSettings)	
(DataBindings)	
(Name)	dataGridView1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColu	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	18
ColumnHeadersHeightSi	AutoSize
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Ready

DataGrid application (step 10)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

CustomerID	CompanyName
*	

DataGridView Tasks
- Choose Data Source: customersBindingSource
- Edit Columns...
- Add Column...
- Enable Adding
- Enable Editing
- Enable Deleting
- Enable Column Reordering
- Dock in parent container
- Add Query...
- Preview Data...

northwindDataSet | customersBindingSource | customersTableAdapter

Properties

dataGridView1 System.Windows.Forms.DataGridV

(ApplicationSettings)	
(DataBindings)	
(Name)	dataGridView1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColur	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	21
ColumnHeadersHeightSi	AutoSize
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Ready

DataGrid application (step 11)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

CustomerID	CompanyName
*	

DataGridView Tasks
Choose Data Source: customersBindingSource
Edit Columns...

Edit Columns

Selected Columns:
- CustomerID
- CompanyName
- ContactName
- ContactTitle
- Address
- City
- Region
- PostalCode
- Country
- Phone
- Fax

Add... Remove

Bound Column Properties

MaxInputLength	32767
ReadOnly	False
Resizable	True
SortMode	Automatic
Data	
DataPropertyName	**CustomerID**
Design	
(Name)	customerIDDataGridViewT
ColumnType	DataGridViewTextBoxColu

(Name)
Indicates the name used in code to identify the object.

OK Cancel

northwindDataSet | customersBindingSource | customersTableAdapter

Properties
dataGridView1 System.Windows.Forms.DataGridV

(ApplicationSettings)	
(DataBindings)	
(Name)	**dataGridView1**
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColu	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	21
ColumnHeadersHeightSi	**AutoSize**
ColumnHeadersVisible	True
Columns	**(Collection)**
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Ready

DataGrid application (step 12)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

Preview Data

Select an object to preview: NorthwindDataSet.Customers.Fill,GetData ()

Target DataSet:

Preview

Parameters:

Name	Type	Value
No parameters are defined on the selected object.		

Results:

CustomerID	CompanyName	ContactName	ContactTitle	Address	City
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Represent...	Obere Str. 57	Berlin
ANATR	Ana Trujillo Emp...	Ana Trujillo	Owner	Avda. de la Con...	México D.F.
ANTON	Antonio Moreno ...	Antonio Moreno	Owner	Mataderos 2312	México D.F.
AROUT	Around the Horn	Thomas Hardy	Sales Represent...	120 Hanover Sq.	London
BERGS	Berglunds snabb...	Christina Berglund	Order Administr...	Berguvsvägen 8	Luleå
BLAUS	Blauer See Delik...	Hanna Moos	Sales Represent...	Forsterstr. 57	Mannheim
BLONP	Blondesddsl pèr...	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg
BOLID	Bólido Comidas p...	Martín Sommer	Owner	C/ Araquil, 67	Madrid
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bou...	Marseille

Columns: 11 Rows: 91

Close

northwindDataSet | customersBindingSource | customersTableAdapter

Properties

dataGridView1 System.Windows.Forms.DataGridV

(ApplicationSettings)	
(DataBindings)	
(Name)	dataGridView1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColur	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	21
ColumnHeadersHeightSi	AutoSize
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Ready

DataGrid application (step 13)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Toolbox
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

Form1.cs [Design]* | Start Page

DataGrid Example

	CustomerID
*	

Search Criteria Builder

Choose an existing query or enter a new query below. A ToolStrip will be added to the form to run the query. To edit an existing query or use stored procedures use the Configure command on the TableAdapter in the DataSet Designer.

Select data source table: NorthwindDataSet.Customers

Select a parameterized query to load data:

New query name: FillBy

Existing query name:

Query Text:

```
SELECT CustomerID, CompanyName, ContactName, ContactTitle, Address, City,
```

Sample: SELECT ColumnName1, ColumnName2 FROM TableName WHERE ColumnName1 = @ParameterName

Query Builder... | OK | Cancel

northwindDataSet | customersBindingSource | customersTableAdapter

Properties

Form1 System.Windows.Forms.Form

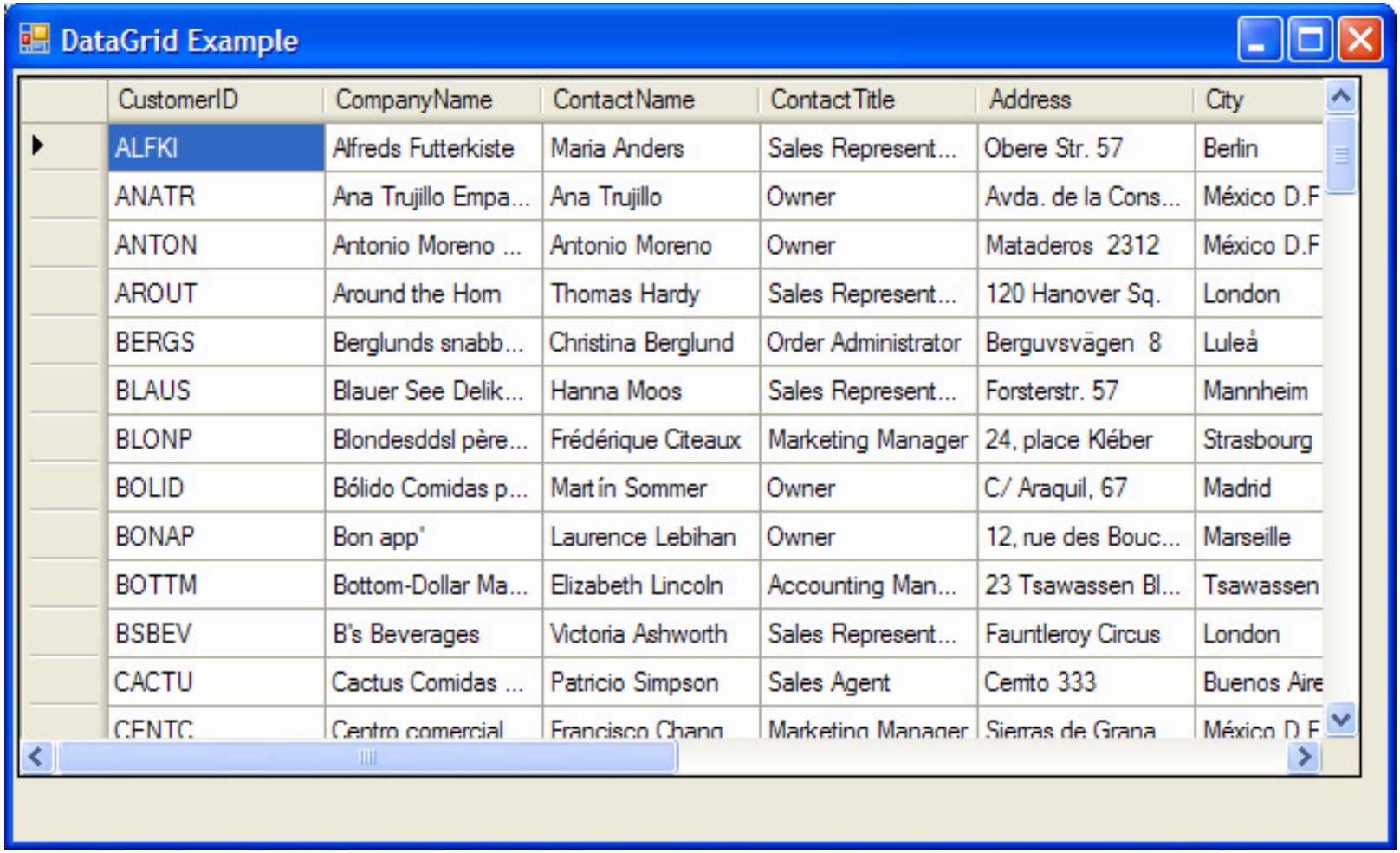
(ApplicationSettings)	
(DataBindings)	
(Name)	Form1
AcceptButton	(none)
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AutoScaleMode	Font
AutoScroll	False
AutoScrollMargin	0; 0
AutoScrollMinSize	0; 0
AutoSize	False
AutoSizeMode	GrowOnly
AutoValidate	EnablePreventFocusChar
BackColor	Control
BackgroundImage	(none)
BackgroundImageLayou	Tile
CancelButton	(none)
CausesValidation	True
ContextMenuStrip	(none)
ControlBox	True
Cursor	Default
DoubleBuffered	False
Enabled	True
Font	Microsoft Sans Serif; 8,25
ForeColor	ControlText
FormBorderStyle	Sizable
HelpButton	False
Icon	(Icon)

(ApplicationSettings)
Maps property settings to an application configuration file.

Toolbox | Server Explorer | Data Sources | Error List

Ready 15, 15 434 x 376

DataGrid application (step 14)



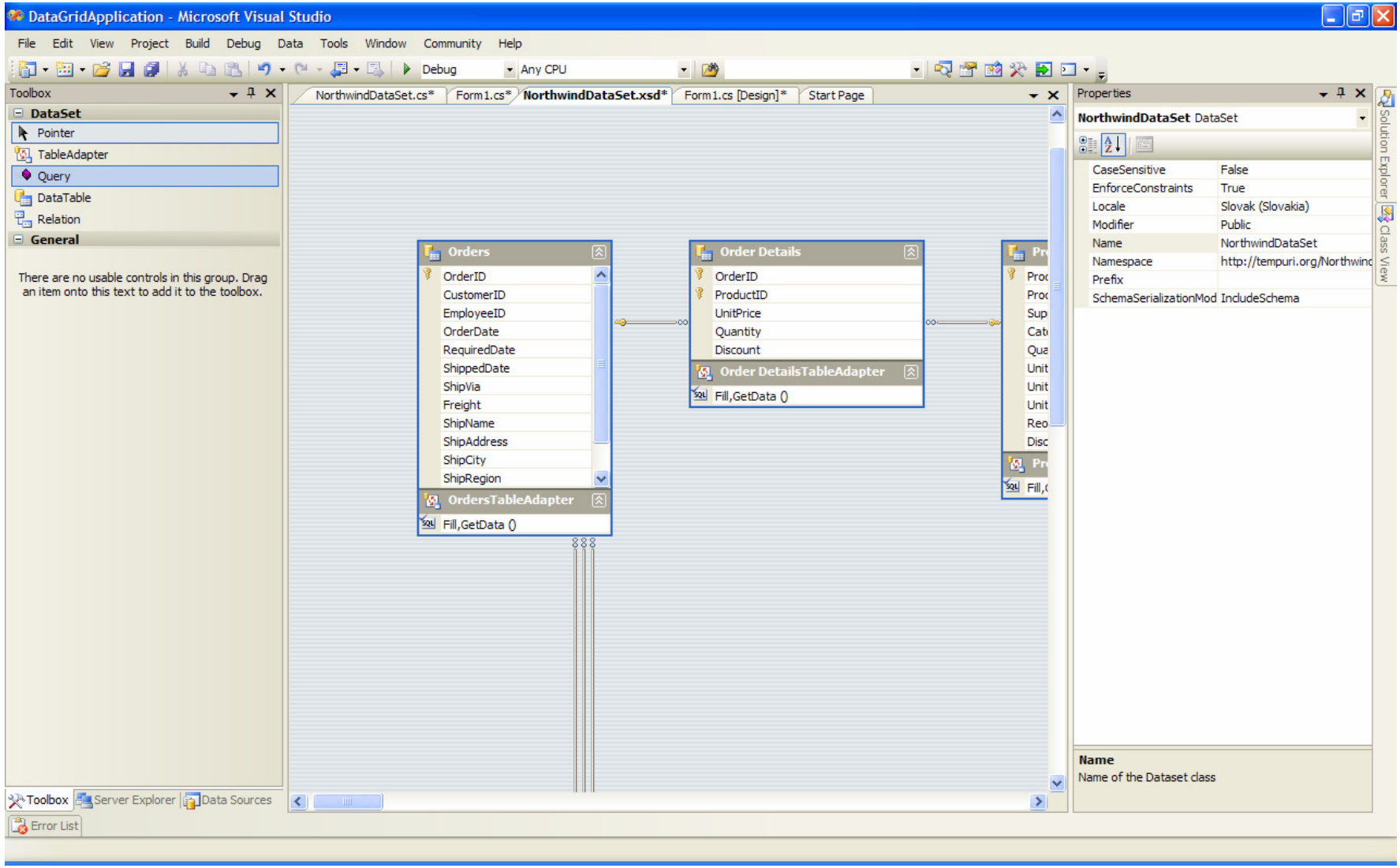
CustomerID	CompanyName	ContactName	ContactTitle	Address	City
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Represent...	Obere Str. 57	Berlin
ANATR	Ana Trujillo Empa...	Ana Trujillo	Owner	Avda. de la Cons...	México D.F
ANTON	Antonio Moreno ...	Antonio Moreno	Owner	Mataderos 2312	México D.F
AROUT	Around the Hom	Thomas Hardy	Sales Represent...	120 Hanover Sq.	London
BERGS	Berglunds snabb...	Christina Berglund	Order Administrator	Berguvsvägen 8	Luleå
BLAUS	Blauer See Delik...	Hanna Moos	Sales Represent...	Forsterstr. 57	Mannheim
BLONP	Blondesddsl père...	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg
BOLID	Bólido Comidas p...	Martín Sommer	Owner	C/ Araquil, 67	Madrid
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bouc...	Marseille
BOTTM	Bottom-Dollar Ma...	Elizabeth Lincoln	Accounting Man...	23 Tsawassen Bl...	Tsawassen
BSBEV	B's Beverages	Victoria Ashworth	Sales Represent...	Fauntleroy Circus	London
CACTU	Cactus Comidas ...	Patricio Simpson	Sales Agent	Cerrito 333	Buenos Aire
CENTC	Centro comercial	Francisco Chang	Marketing Manager	Sierras de Grana	México D.F

Data binding

Data binding allows us to quickly create user interface based on data retrieved from database. Data are represented by common database components to form GUI.

When DataSet is created you can easily check data structure inside Visual Studio .NET, create user interface based on grid or detailed form.

DataSet diagram



DataBinding application (step 1)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Data Sources

- NorthwindDataSet
  - Alphabetical list of products
  - Categories
  - Category Sales for 1997
  - Current Product List
  - Customer and Suppliers by City
  - CustomerCustomerDemo
  - CustomerDemographics
  - Customers
  - Employees
  - EmployeeTerritories
  - Invoices
  - Order Details
  - Order Details Extended
  - Order Subtotals
  - Orders
  - Orders Qry
  - Product Sales for 1997
  - Products
  - Products Above Average Price
  - Products by Category
  - Quarterly Orders
  - Region
  - Sales by Category
  - Sales Totals by Amount
  - Shippers
  - Summary of Sales by Quarter
  - Summary of Sales by Year
  - Suppliers
  - Territories

Toolbox | Server Explorer | Data Sources

Error List

Start Page | NorthwindDataSet.cs | Form1.cs | NorthwindDataSet.xsd | Form1.cs [Design]

DataBinding Example

northwindDataSet | customersBindingSource | customersTableAdapter

Properties

Form1 System.Windows.Forms.Form

FormBorderStyle	Sizable
HelpButton	False
Icon	(Icon)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(Default)
Localizable	False
Location	0; 0
Locked	False
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0; 0
MinimizeBox	True
MinimumSize	0; 0
Opacity	100%
Padding	0; 0; 0; 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	**640; 448**
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**DataBinding Example**
TopMost	False
TransparencyKey	

Text
The text associated with the control.

Solution Explorer | Class View

Item(s) Saved

DataBinding application (step 2)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Window Community Help

Debug | Any CPU

Data Sources

- NorthwindDataSet
  - Alphabetical list of products
  - Categories
  - Category Sales for 1997
  - Current Product List
  - Customer and Suppliers by City
  - CustomerCustomerDemo
  - CustomerDemographics
  - Customers
    - DataGridView
    - Details
    - [None]
    - Customize...
    - Region
    - PostalCode
    - Country
    - Phone
    - Fax
    - CustomerCustomerDemo
    - Orders
  - Employees
  - EmployeeTerritories
  - Invoices
  - Order Details
  - Order Details Extended
  - Order Subtotals
  - Orders
  - Orders Qry
  - Product Sales for 1997
  - Products
  - Products Above Average Price
  - Products by Category
  - Quarterly Orders
  - Region

Start Page | NorthwindDataSet.cs | Form1.cs | NorthwindDataSet.xsd | Form1.cs [Design]

DataBinding Example

northwindDataSet | customersBindingSource | customersTableAdapter

Properties

Form1 System.Windows.Forms.Form

FormBorderStyle	Sizable
HelpButton	False
Icon	(Icon)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(Default)
Localizable	False
Location	0; 0
Locked	False
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0; 0
MinimizeBox	True
MinimumSize	0; 0
Opacity	100%
Padding	0; 0; 0; 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	**640; 448**
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**DataBinding Example**
TopMost	False
TransparencyKey	
UseWaitCursor	False
WindowState	Normal

Text
The text associated with the control.

Toolbox | Server Explorer | Data Sources | Error List

Item(s) Saved

DataBinding application (step 3)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Data Sources

- NorthwindDataSet
  - Alphabetical list of products
  - Categories
  - Category Sales for 1997
  - Current Product List
  - Customer and Suppliers by City
  - CustomerCustomerDemo
  - CustomerDemographics
  - Customers
    - CustomerID
    - CompanyName
    - ContactName
    - ContactTitle
    - Address
    - City
    - Region
    - PostalCode
    - Country
    - Phone
    - Fax
    - CustomerCustomerDemo
    - Orders
  - Employees
  - EmployeeTerritories
  - Invoices
  - Order Details
  - Order Details Extended
  - Order Subtotals
  - Orders
  - Orders Qry
  - Product Sales for 1997
  - Products
  - Products Above Average Price
  - Products by Category
  - Quarterly Orders
  - Region

Toolbox | Server Explorer | Data Sources | Error List

Start Page | NorthwindDataSet.cs | Form1.cs | NorthwindDataSet.xsd | Form1.cs [Design]

DataBinding Example

Customer ID:
Company Name:
Contact Name:
Contact Title:
Address:
City:
Region:
Postal Code:
Country:
Phone:
Fax:

northwindDataSet | customersBindingSource | customersTableAdapter

Properties

Form1 System.Windows.Forms.Form

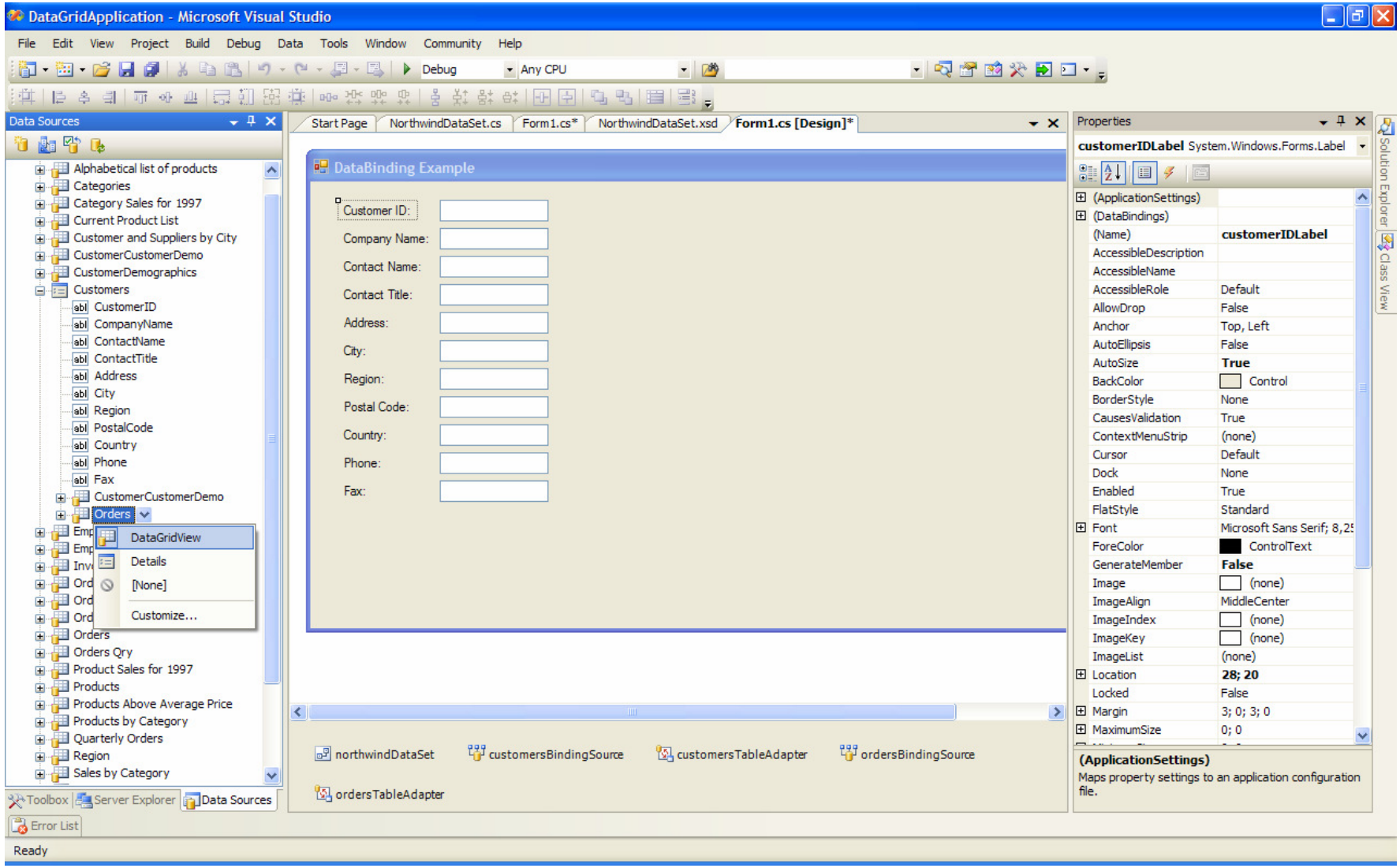
ForeColor	ControlText
FormBorderStyle	Sizable
HelpButton	False
Icon	(Icon)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(Default)
Localizable	False
Location	0; 0
Locked	False
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0; 0
MinimizeBox	True
MinimumSize	0; 0
Opacity	100%
Padding	0; 0; 0; 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	**640; 448**
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**DataBinding Example**
TopMost	False
TransparencyKey	
UseWaitCursor	False

Text
The text associated with the control.

Solution Explorer | Class View

Ready | 15, 15 | 640 x 448

DataBinding application (step 4)



DataBinding application (step 5)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Window Community Help

Debug | Any CPU

Data Sources

- NorthwindDataSet
  - Alphabetical list of products
  - Categories
  - Category Sales for 1997
  - Current Product List
  - Customer and Suppliers by City
  - CustomerCustomerDemo
  - CustomerDemographics
  - Customers
    - CustomerID
    - CompanyName
    - ContactName
    - ContactTitle
    - Address
    - City
    - Region
    - PostalCode
    - Country
    - Phone
    - Fax
    - CustomerCustomerDemo
    - Orders
  - Employees
  - EmployeeTerritories
  - Invoices
  - Order Details
  - Order Details Extended
  - Order Subtotals
  - Orders
  - Orders Qry
  - Product Sales for 1997
  - Products
  - Products Above Average Price
  - Products by Category
  - Quarterly Orders
  - Region

Toolbox | Server Explorer | Data Sources | Error List

Start Page | NorthwindDataSet.cs | Form1.cs | NorthwindDataSet.xsd | Form1.cs [Design]

DataBinding Example

Customer ID:
Company Name:
Contact Name:
Contact Title:
Address:
City:
Region:
Postal Code:
Country:
Phone:
Fax:

OrderID	CustomerID	EmployeeID	OrderDate

northwindDataSet | customersBindingSource | customersTableAdapter | ordersBindingSource | ordersTableAdapter | ordersBindingSource1

Properties

ordersDataGridView System.Windows.Forms.Dat

(ApplicationSettings)	
(DataBindings)	
(Name)	ordersDataGridView
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowUserToAddRows	True
AllowUserToDeleteRows	True
AllowUserToOrderColum	False
AllowUserToResizeColur	True
AllowUserToResizeRows	True
AlternatingRowsDefault	DataGridViewCellStyle { }
Anchor	Top, Left
AutoSizeColumnsMode	None
AutoSizeRowsMode	None
BackgroundColor	AppWorkspace
BorderStyle	FixedSingle
CausesValidation	True
CellBorderStyle	Single
ClipboardCopyMode	EnableWithAutoHeaderTe
ColumnHeadersBorderS	Raised
ColumnHeadersDefaultC	DataGridViewCellStyle { B
ColumnHeadersHeight	23
ColumnHeadersHeightSi	EnableResizing
ColumnHeadersVisible	True
Columns	(Collection)
ContextMenuStrip	(none)
Cursor	Default
DataMember	

(ApplicationSettings)
Maps property settings to an application configuration file.

Solution Explorer | Class View

Ready | 246, 21 | 438 x 272

DataBinding application (step 6)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Window Community Help

Debug | Any CPU

Toolbox
- DataGridApplication Components
- All Windows Forms
- Common Controls
- Containers
- Menus & Toolbars
- Data
  - Pointer
  - DataSet
  - DataGridView
  - BindingSource
  - BindingNavigator
  - ReportViewer
- Components
- Printing
- Dialogs
- Crystal Reports
- General

There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

Start Page | NorthwindDataSet.cs | Form1.cs* | NorthwindDataSet.xsd | Form1.cs [Design]*

DataBinding Example

0 of {0}

Customer ID:
Company Name:
Contact Name:
Contact Title:
Address:
City:
Region:
Postal Code:
Country:
Phone:
Fax:

OrderID	CustomerID	EmployeeID	OrderDate

northwindDataSet | customersBindingSource | customersTableAdapter | ordersBindingSource
ordersTableAdapter | ordersBindingSource1 | bindingNavigator1

Properties
bindingNavigator1 System.Windows.Forms.Bindin

Enabled	True
Font	Tahoma; 8,25pt
GenerateMember	True
GripMargin	2; 2; 2; 2
GripStyle	Visible
ImageScalingSize	16; 16
ImeMode	NoControl
Items	(Collection)
LayoutStyle	HorizontalStackWithOver
Location	0; 0
Locked	False
Margin	0; 0; 0; 0
MaximumSize	0; 0
MinimumSize	0; 0
Modifiers	Private
MoveFirstItem	bindingNavigatorMov
MoveLastItem	bindingNavigatorMov
MoveNextItem	bindingNavigatorMov
MovePreviousItem	bindingNavigatorMov
Padding	0; 0; 1; 0
PositionItem	bindingNavigatorPosi
RenderMode	ManagerRenderMode
RightToLeft	No
ShowItemToolTips	True
Size	777; 25
Stretch	False
TabIndex	23
TabStop	False
Tag	
Text	bindingNavigator1

Text
The text associated with the control.

Toolbox | Server Explorer | Data Sources | Error List

Ready

DataBinding application (step 7)

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Window Community Help

Debug | Any CPU

Data Sources

- NorthwindDataSet
  - Alphabetical list of products
  - Categories
  - Category Sales for 1997
  - Current Product List
  - Customer and Suppliers by City
  - CustomerCustomerDemo
  - CustomerDemographics
  - Customers
    - CustomerID
    - CompanyName
    - ContactName
    - ContactTitle
    - Address
    - City
    - Region
    - PostalCode
    - Country
    - Phone
    - Fax
    - CustomerCustomerDemo
    - Orders
  - Employees
  - EmployeeTerritories
  - Invoices
  - Order Details
  - Order Details Extended
  - Order Subtotals
  - Orders
  - Orders Qry
  - Product Sales for 1997
  - Products
  - Products Above Average Price
  - Products by Category
  - Quarterly Orders
  - Region

Toolbox | Server Explorer | Data Sources | Error List

Start Page | NorthwindDataSet.cs | Form1.cs* | NorthwindDataSet.xsd | Form1.cs [Design]*

DataBinding Example

0 of {0}

Customer ID:
Company Name:
Contact Name:
Contact Title:
Address:
City:
Region:
Postal Code:
Country:
Phone:
Fax:

OrderID	CustomerID	EmployeeID	OrderDate

northwindDataSet | customersBindingSource | customersTableAdapter | ordersTableAdapter | ordersBindingSource1 | bindingNavigator1

Properties

bindingNavigator1 System.Windows.Forms.Bindin

(ApplicationSettings)	
(DataBindings)	
(Name)	bindingNavigator1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AddNewItem	bindingNavigatorAddI
AllowDrop	False
AllowItemReorder	False
AllowMerge	True
Anchor	Top, Left
AutoSize	True
BackColor	Control
BackgroundImage	(none)
BackgroundImageLayou	Tile
BindingSource	ersBindingSource
CanOverflow	(none)
ContextMenuStri	customersBindingSource
CountItem	ordersBindingSource1
CountItemFormat	of {0}
DeleteItem	bindingNavigatorDele
Dock	Top
Enabled	True
Font	Tahoma; 8,25pt
GenerateMember	True
GripMargin	2; 2; 2; 2
GripStyle	Visible
ImageScalingSize	16; 16
ImeMode	NoControl
Items	(Collection)

BindingSource
Specifies support for transacted initialization.

Solution Explorer | Class View

Ready

DataBinding application (step 8)

DataBinding Example

6 of 91

Current position

Customer ID: BLAUS
Company Name: Blauer See Delikate
Contact Name: Hanna Moos
Contact Title: Sales Representati
Address: Forsterstr. 57
City: Mannheim
Region:
Postal Code: 68306
Country: Germany
Phone: 0621-08460
Fax: 0621-08924

OrderID	CustomerID	EmployeeID	OrderDate
10501	BLAUS	9	9. 4. 1997
10509	BLAUS	4	17. 4. 1997
10582	BLAUS	3	27. 6. 1997
10614	BLAUS	8	29. 7. 1997
10853	BLAUS	9	27. 1. 1998
10956	BLAUS	6	17. 3. 1998
11058	BLAUS	9	29. 4. 1998
*			

Working with SQL Server in VS .NET

DataGridApplication - Microsoft Visual Studio

File Edit View Project Build Debug Data Query Designer Tools Window Community Help

Debug | Any CPU

Change Type

Server Explorer

- Data Connections
  - misik.Northwind.dbo
    - Database Diagrams
    - Tables
      - Categories
      - CustomerCustomerDemo
      - CustomerDemographics
      - Customers
        - CustomerID
        - CompanyName
        - ContactName
        - ContactTitle
        - Address
        - City
        - Region
        - PostalCode
        - Country
        - Phone
        - Fax
      - Employees
      - EmployeeTerritories
      - Order Details
      - Orders
      - Products
      - Region
      - Shippers
      - Suppliers
      - Territories
    - Views
    - Stored Procedures
    - Functions
    - Synonyms
    - Types
    - Assemblies

Customers: Que...isik.Northwind) | dbo.Customers:...isik.Northwind) | Start Page | NorthwindDataSet.cs | Form1.cs

CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Represent...	Obere Str. 57	Berlin	NULL
ANATR	Ana Trujillo Emp...	Ana Trujillo	Owner	Avda. de la Con...	México D.F.	NULL
ANTON	Antonio Moreno ...	Antonio Moreno	Owner	Mataderos 2312	México D.F.	NULL
AROUT	Around the Horn	Thomas Hardy	Sales Represent...	120 Hanover Sq.	London	NULL
BERGS	Berglunds snabb...	Christina Berglund	Order Administr...	Berguvsvägen 8	Luleå	NULL
BLAUS	Blauer See Delik...	Hanna Moos	Sales Represent...	Forsterstr. 57	Mannheim	NULL
BLONP	Blondesddsl pèr...	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg	NULL
BOLID	Bólido Comidas p...	Martín Sommer	Owner	C/ Araquil, 67	Madrid	NULL
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bou...	Marseille	NULL
BOTTM	Bottom-Dollar M...	Elizabeth Lincoln	Accounting Man...	23 Tsawassen Bl...	Tsawassen	BC
BSBEV	B's Beverages	Victoria Ashworth	Sales Represent...	Fauntleroy Circus	London	NULL
CACTU	Cactus Comidas ...	Patricio Simpson	Sales Agent	Cerrito 333	Buenos Aires	NULL
CENTC	Centro comercial...	Francisco Chang	Marketing Manager	Sierras de Grana...	México D.F.	NULL
CHOPS	Chop-suey Chin...	Yang Wang	Owner	Hauptstr. 29	Bern	NULL
COMMI	Comércio Mineiro	Pedro Afonso	Sales Associate	Av. dos Lusíada...	Sao Paulo	SP
CONSH	Consolidated Hol...	Elizabeth Brown	Sales Represent...	Berkeley Garden...	London	NULL
DRACD	Drachenblut Deli...	Sven Ottlieb	Order Administr...	Walserweg 21	Aachen	NULL
DUMON	Du monde entier	Janine Labrune	Owner	67, rue des Cinq...	Nantes	NULL
EASTC	Eastern Connect...	Ann Devon	Sales Agent	35 King George	London	NULL
ERNSH	Ernst Handel	Roland Mendel	Sales Manager	Kirchgasse 6	Graz	NULL
FAMIA	Familia Arquibaldo	Aria Cruz	Marketing Assist...	Rua Orós, 92	Sao Paulo	SP
FISSA	FISSA Fabrica In...	Diego Roel	Accounting Man...	C/ Moralzarzal, 86	Madrid	NULL
FOLIG	Folies gourmandes	Martine Rancé	Assistant Sales ...	184, chaussée d...	Lille	NULL
FOLKO	Folk och fä HB	Maria Larsson	Owner	Åkergatan 24	Bräcke	NULL
FRANK	Frankenversand	Peter Franken	Marketing Manager	Berliner Platz 43	München	NULL

1 of 91

Properties

[Qry] Query

(Name)	Query
Database Name	Northwind
Destination Table	
Distinct Values	No
GROUP BY Extension	<None>
Output All Columns	Yes
Query Parameter List	No parameters have been sp
Server Name	misik
SQL Comment	
Top Specification	No

(Name)

Toolbox | Server Explorer | Data Sources | Error List

Ready

Using MySQL in Visual Studio .NET

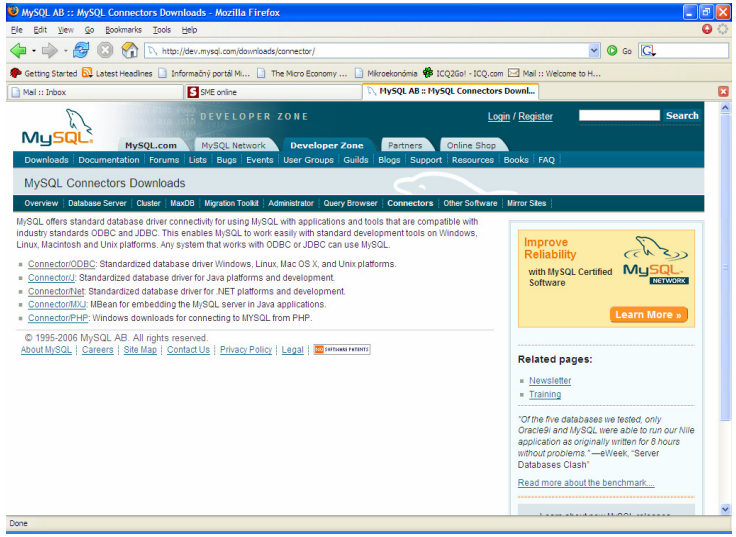
Using MySQL from Visual Studio .NET requires little bit different techniques to retrieving and manipulating data than was presented for Microsoft SQL Server.

In the first step is necessary to prepare components to connecting database from Visual Studio workspace. The connector for .NET is available on MySQL web site.

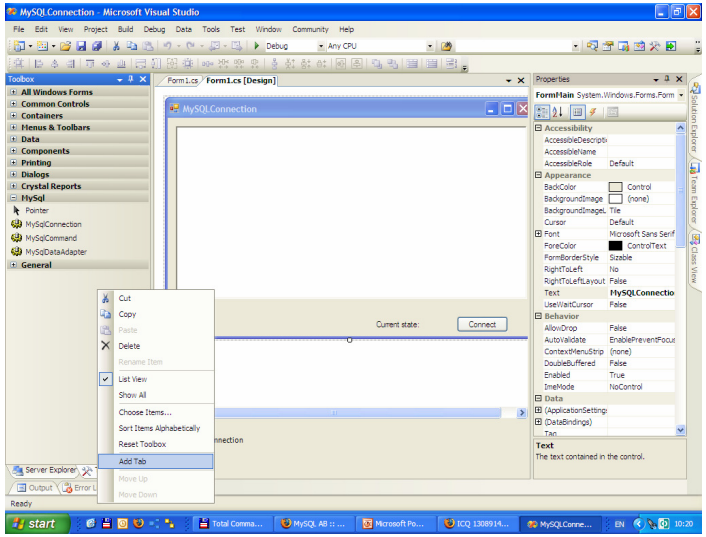
Data manipulation presented in examples is mostly implemented inside application sources manually. Hence is easy to use demonstrated procedures in other applications.

Installing MySQL .NET connector

MySQL is using standardized connector for .NET platform. To develop application on Windows is necessary to download and install the connector and then to register it in the Visual Studio.



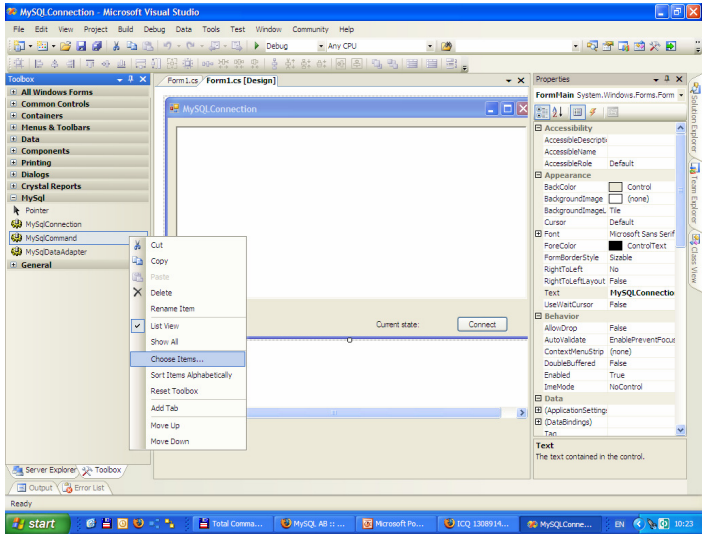
In the Visual Studio choose *Toolbox* (if not visible use menu *View-Toolbox*), right click into it and choose *Add Tab* from the menu. Inside the created box write e.g. *MySQL* as the caption for the new tab.

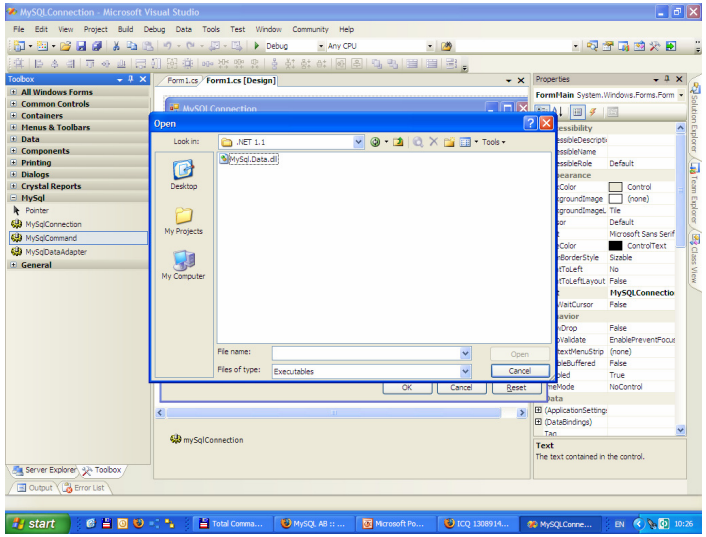


Registering .NET connector

Right click into the new tab from Toolbox and click *Choose Item* menu. After couple of seconds *Choose Toolbox Items* dialog appear and you can easily *Browse .dll* file with the .NET connector (in our case in *c:\Program Files\MySQL\MySQL Connector Net 1.0.7\bin\.NET 1.1.\MySQL.Data.dll*). Finally three new items appear inside the Toolbox's MySQL tab:

- *MySqlConnection*
- *MySqlCommand*
- *MySqlDataAdapter*

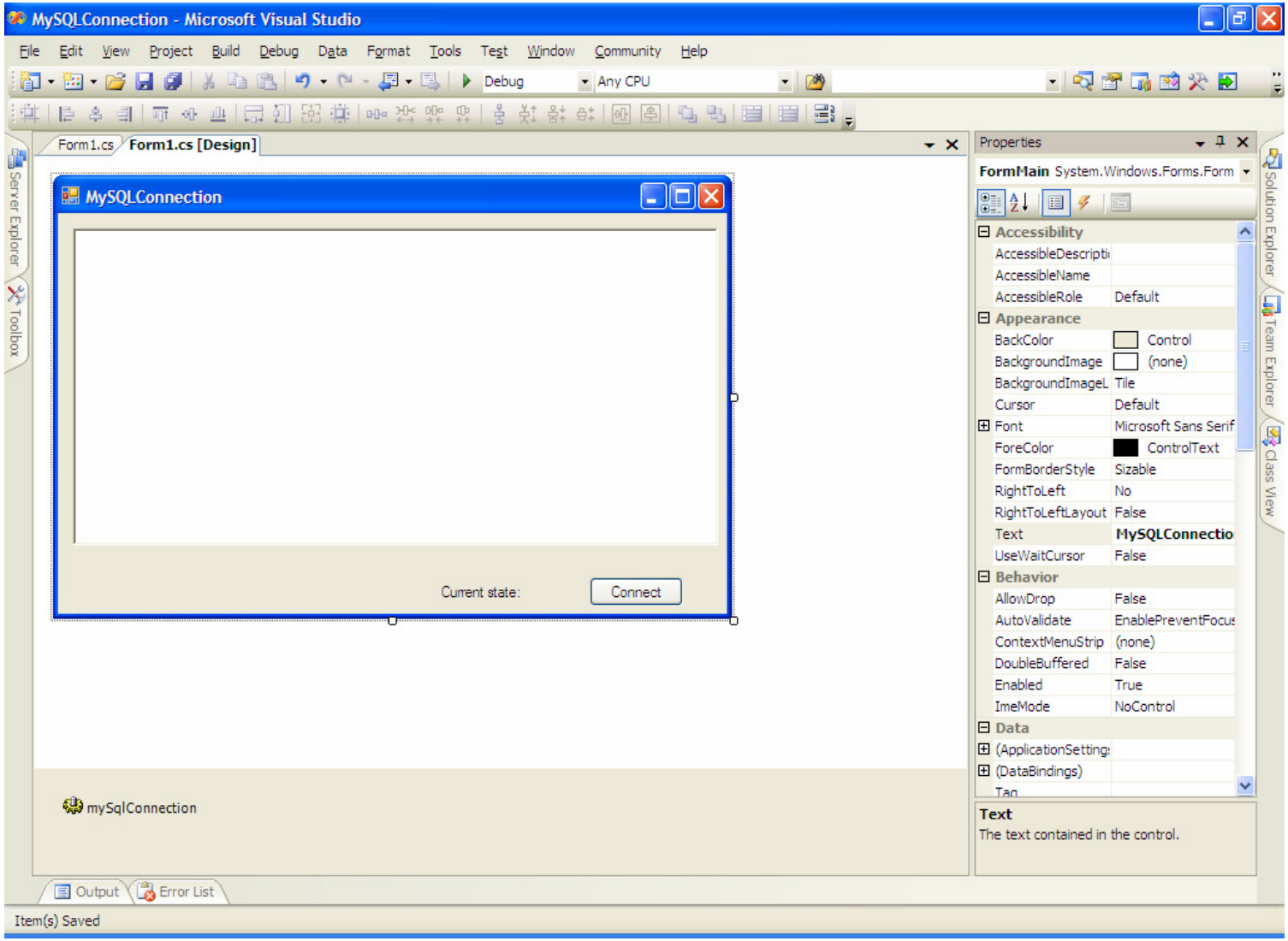




Examples using MySQL

- **MySQLConnection** – simple windows application using **MySqlConnenction** object and demonstrating basic windows based programming techniques such as event oriented programming.
- **MySQLApplication** - simple windows application demonstrating how to use **DataGridView** component with MySQL.
- **FormElementBinding** – example of binding form element by using MySQL and demonstration of a Master-Detail relationship.
- **MySQLTableEditor** – complex MySQL example (official example from MySQL documentation) demonstrating basic features of the .NET connector.
- **MySQLModifyData** – application shows how to use INSERT, UPDATE and DELETE statements in C# code. Also concept of transactional processing is introduced here.

MySQLConnection example (1)



MySQLConnection example (2)

MySQLConnection - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

MySQLConnection.FormMain | buttonConnect_Click(object sender, EventArgs e)

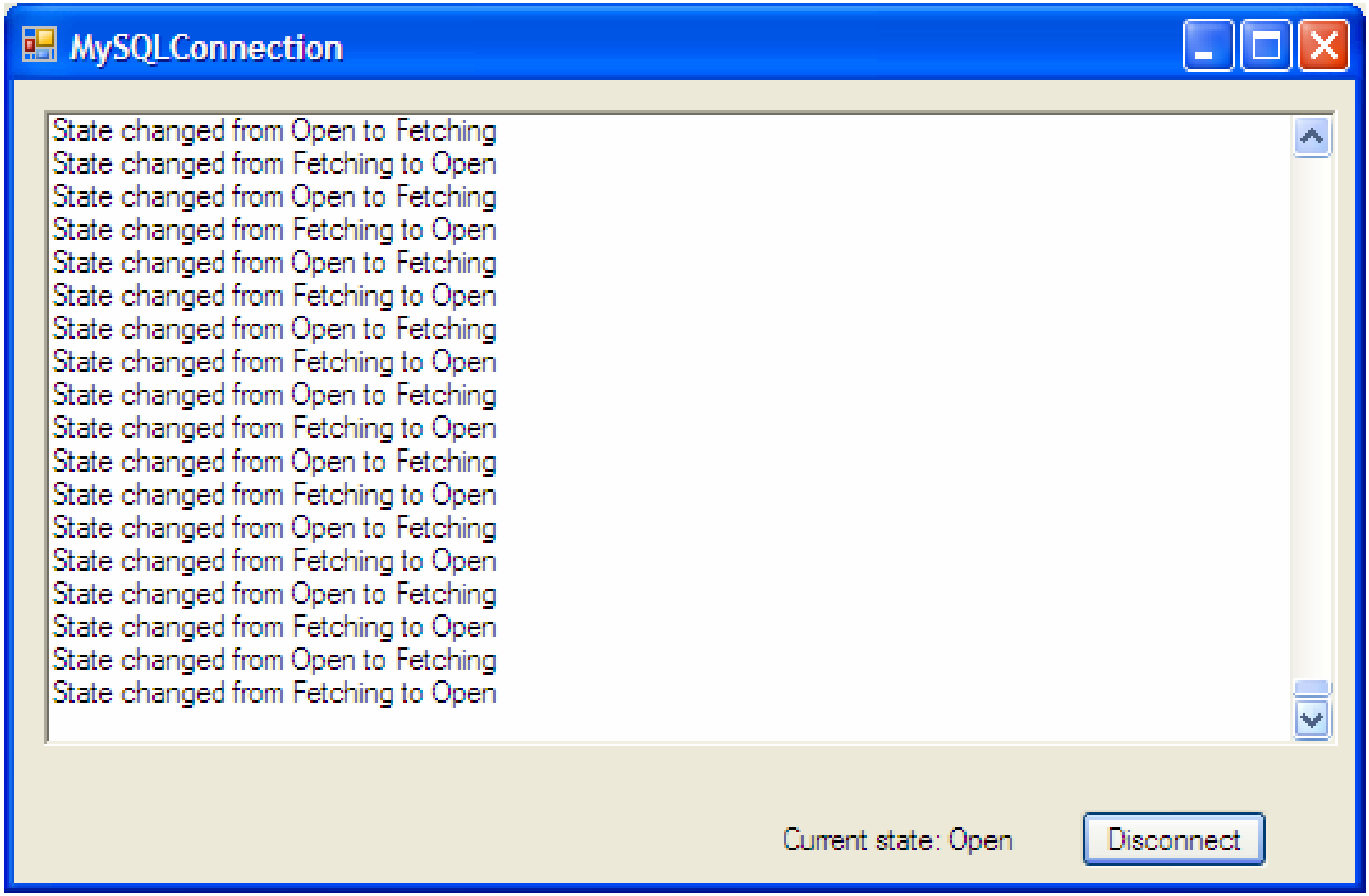
```
namespace MySQLConnection {
    public partial class FormMain : Form {
        public FormMain () {
            InitializeComponent ();
        }

        private void FormMain_Load (object sender, EventArgs e) {

        }

        private void buttonConnect_Click (object sender, EventArgs e) {
            if (mySqlConnection.State == ConnectionState.Closed) {
                richTextBoxLog.Clear ();
                mySqlConnection.Open ();
                buttonConnect.Text = "Disconnect";
            } else {
                richTextBoxLog.Clear ();
                mySqlConnection.Close ();
                buttonConnect.Text = "Connect";
            }
            labelState.Text = "Current state: " + mySqlConnection.State.ToString ();
        }

        private void mySqlConnection_StateChange (object sender, StateChangeEventArgs e
            richTextBoxLog.AppendText (
                "State changed from " + e.OriginalState +
                " to " + e.CurrentState + "\n");
        }
    }
}
```

Properties

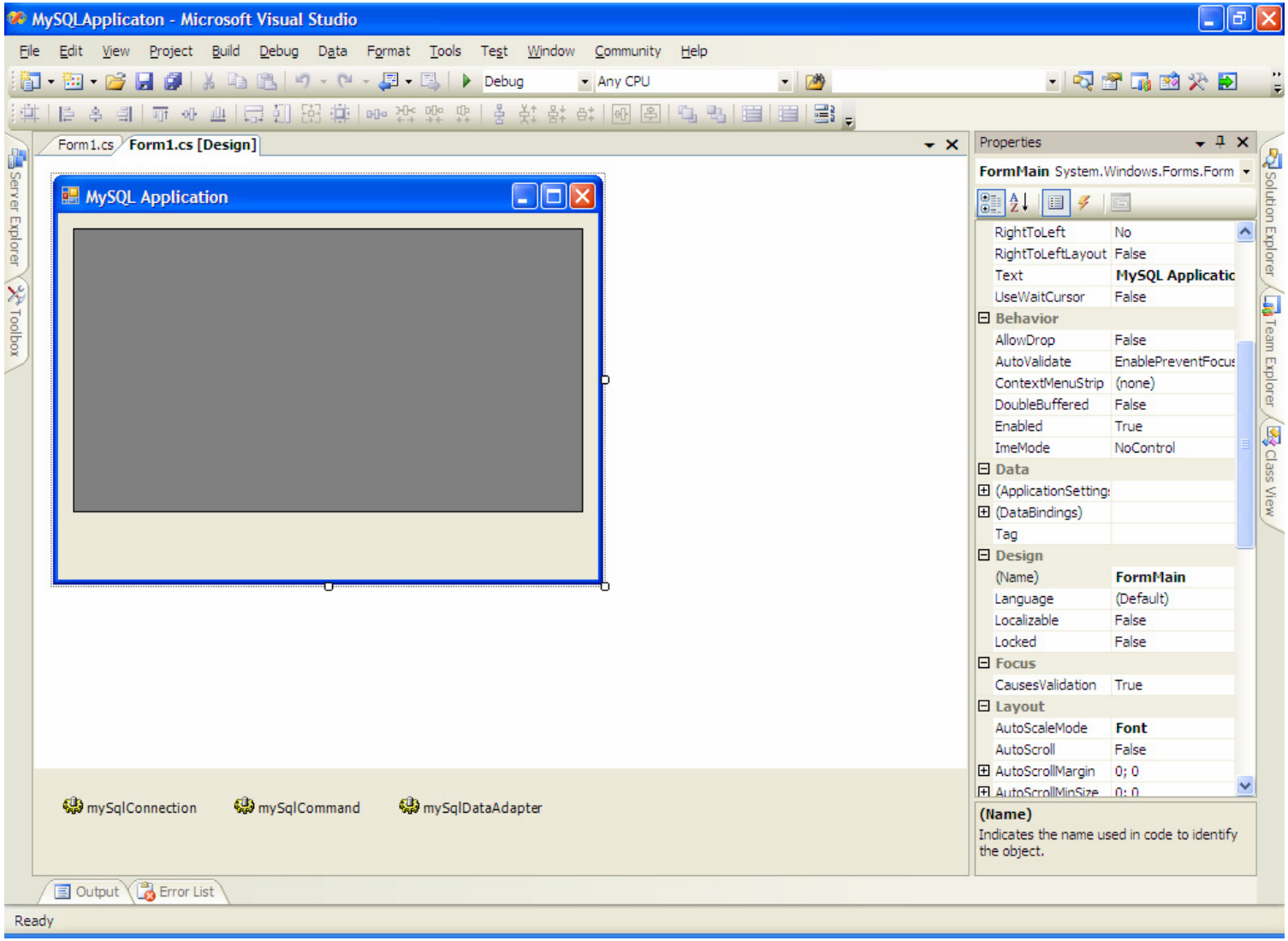
Output | Error List

Item(s) Saved | Ln 28 | Col 14 | Ch 5 | INS

MySQLConnection example (3)



MySQLApplication example (1)



MySQLApplication example (2)

MySQLApplicaton - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

MySQLApplicaton.FormMain | FormMain()

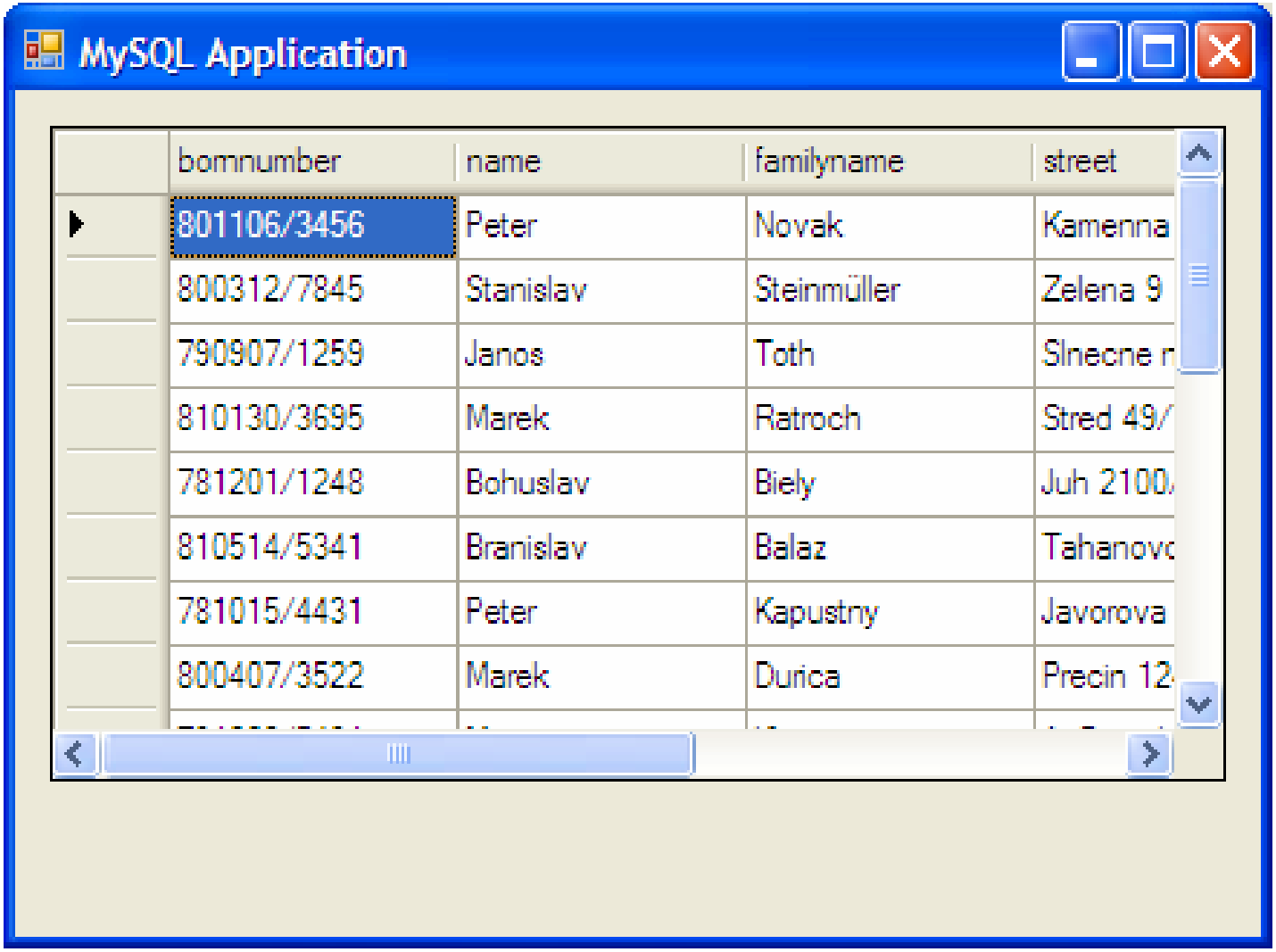
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace MySQLApplicaton {
    public partial class FormMain : Form {
        public FormMain () {
            InitializeComponent ();

            DataTable dataTable = new DataTable ();

            mySqlConnection.Open ();
            mySqlDataAdapter.Fill (dataTable);
            mySqlConnection.Close ();

            dataGridView1.DataSource = dataTable;

        }
    }
}
```

Properties

Output | Error List

Ready | Ln 10 | Col 43 | Ch 40 | INS

MySQLApplication example (3)



bornumber	name	familyname	street
801106/3456	Peter	Novak	Kamenna
800312/7845	Stanislav	Steinmüller	Zelena 9
790907/1259	Janos	Toth	Slnecne r
810130/3695	Marek	Ratroch	Stred 49/
781201/1248	Bohuslav	Biely	Juh 2100/
810514/5341	Branislav	Balaz	Tahanovc
781015/4431	Peter	Kapustny	Javorova
800407/3522	Marek	Durica	Precin 12

FormElementBinding example (1)

FormElementBinding - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Test Window Community Help

Debug | Any CPU

Program.cs | Form1.cs | Form1.cs [Design]

Data Bindings and Master-Detail

Name:

Surname:

Courses

Teachers

mySqlConnection | mySqlCommand | mySqlDataAdapter

Properties

FormMain System.Windows.Forms.Form

Accessibility
AccessibleDescripti
AccessibleName
AccessibleRole Default

Appearance
BackColor Control
BackgroundImage (none)
BackgroundImageL Tile
Cursor Default
Font Microsoft Sans Serif
ForeColor ControlText
FormBorderStyle Sizable
RightToLeft No
RightToLeftLayout False
Text Data Bindings ar
UseWaitCursor False

Behavior
AllowDrop False
AutoValidate EnablePreventFocus
ContextMenuStrip (none)
DoubleBuffered False
Enabled True
ImeMode NoControl

Data
(ApplicationSetting
(DataBindings)
Tag

Text
The text contained in the control.

Output | Error List

Item(s) Saved

start | Total Comm... | Mail :: Inbox... | Microsoft Po... | ICQ 130891... | FormElemen... | EN | 11:32

FormElementBinding example (2)

FormElementBinding - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Program.cs | Form1.cs | Form1.cs [Design]

FormElementBinding.FormMain | FormMain()

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace FormElementBinding {

    public partial class FormMain : Form {
        private DataTable dataTable = new DataTable ();
        public FormMain () {
            InitializeComponent ();

            try {
                mySqlConnection.Open ();
                mySqlDataAdapter.Fill (dataTable);
                mySqlConnection.Close ();
            } catch (Exception e) {
                MessageBox.Show ("Error: " + e.Message);
            }

            dataGridViewTeachers.DataSource = dataTable;
        }

        private void FormMain_Load (object sender, EventArgs e) {

        }
```

Ready | Ln 25 | Col 57 | Ch 48 | INS

FormElementBinding example (3)

FormElementBinding - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Program.cs | Form1.cs | Form1.cs [Design]

FormElementBinding.FormMain | FormMain()

```
        }

        private void dataGridViewTeachers_RowEnter (object sender, DataGridViewCellEven
            try {
                //MessageBox.Show ("Idx: " + e.RowIndex.ToString ());
                textBoxName.Text = dataGridViewTeachers.Rows[e.RowIndex].Cells[1].Value
                textBoxSurname.Text = dataGridViewTeachers.Rows[e.RowIndex].Cells[2].Va

                String teacherID = dataGridViewTeachers.Rows[e.RowIndex].Cells[0].Value

                // Detail goes to DbGrid
                MySqlCommand coursesCmd = mySqlConnection.CreateCommand ();
                coursesCmd.CommandText =
                    "SELECT code, name " +
                    "FROM subject " +
                    "WHERE gestor = '" + teacherID + "'";

                MySqlDataAdapter coursesDataAdapter = new MySqlDataAdapter ();
                coursesDataAdapter.SelectCommand = coursesCmd;

                DataTable coursesData = new DataTable ();
                coursesDataAdapter.Fill (coursesData);

                dataGridViewCourses.DataSource = coursesData;

            } catch (Exception ex) {
                MessageBox.Show (ex.Message);
            }
        }
```

Ready | Ln 25 | Col 57 | Ch 48 | INS

FormElementBinding example (4)

Data Bindings and Master-Detail

Name: Rudolf

Surname: Kodnar

Courses

	code	name
▶	P102	Algebra
	P103	Matematicka anal...
	P202	Matematicka anal...
	P303	Matematicka anal...
*		

Teachers

	teacherid	name	familyname	department
	KI001	Stefan	Kovalik	KI
▶	KMM02	Rudolf	Kodnar	KMM
	KMM03	Stanislav	Paluch	KMM
	KI002	Peter	Varsa	KI

MySQLTableEditor example (1)

TableEditor - Microsoft Visual Studio

File Edit View Project Build Debug Data Format Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

Form1

Server:

Connect

User Id:

Password:

Databases

Tables

Update

Properties

Form1 System.Windows.Forms.Form

Accessibility	
AccessibleDescripti	
AccessibleName	
AccessibleRole	Default
Appearance	
BackColor	Control
BackgroundImage	(none)
BackgroundImageL	Tile
Cursor	Default
Font	Microsoft Sans Serif
ForeColor	ControlText
FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
Text	Form1
UseWaitCursor	False
Behavior	
AllowDrop	False
AutoValidate	EnablePreventFocus
ContextMenuStrip	(none)
DoubleBuffered	False
Enabled	True
ImeMode	NoControl
Data	
(ApplicationSetting	
(DataBindings)	
Tag	

Text
The text contained in the control.

Output | Error List

Ready

MySQLTableEditor example (2)

TableEditor - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs* | Form1.cs [Design]*

TableEditor.Form1 | label1

```
        private void connectBtn_Click(object sender, System.EventArgs e)
        {
            if (conn != null)
                conn.Close();

            string connStr = String.Format("server={0};user id={1}; password={2}; datab
                server.Text, userid.Text, password.Text );

            try
            {
                conn = new MySqlConnection( connStr );
                conn.Open();

                GetDatabases();
            }
            catch (MySqlException ex)
            {
                MessageBox.Show( "Error connecting to the server: " + ex.Message );
            }
        }

        private void GetDatabases()
        {
            MySqlDataReader reader = null;

            MySqlCommand cmd = new MySqlCommand("SHOW DATABASES", conn);
            try
            {
                reader = cmd.ExecuteReader();
                databaseList.Items.Clear();
```

Output | Error List

Ready | Ln 34 | Col 51 | Ch 48 | INS

MySQLTableEditor example (3)

TableEditor - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs* | Form1.cs [Design]*

TableEditor.Form1 | label1

```
        private void GetDatabases()
        {
            MySqlDataReader reader = null;

            MySqlCommand cmd = new MySqlCommand("SHOW DATABASES", conn);
            try
            {
                reader = cmd.ExecuteReader();
                databaseList.Items.Clear();
                while (reader.Read())
                {
                    databaseList.Items.Add( reader.GetString(0) );
                }
            }
            catch (MySqlException ex)
            {
                MessageBox.Show("Failed to populate database list: " + ex.Message );
            }
            finally
            {
                if (reader != null) reader.Close();
            }
        }

        private void databaseList_SelectedIndexChanged(object sender, System.EventArgs
        {
            MySqlDataReader reader = null;

            conn.ChangeDatabase( databaseList.SelectedItem.ToString() );
```

Output | Error List

Ready | Ln 34 | Col 51 | Ch 48 | INS

MySQLTableEditor example (4)

TableEditor - Microsoft Visual Studio

Form1.cs* | Form1.cs [Design]*

TableEditor.Form1 | label1

```
        private void databaseList_SelectedIndexChanged(object sender, System.EventArgs
        {
            MySqlDataReader reader = null;

            conn.ChangeDatabase( databaseList.SelectedItem.ToString() );

            MySqlCommand cmd = new MySqlCommand("SHOW TABLES", conn);
            try
            {
                reader = cmd.ExecuteReader();
                tables.Items.Clear();
                while (reader.Read())
                {
                    tables.Items.Add( reader.GetString(0) );
                }
            }
            catch (MySqlException ex)
            {
                MessageBox.Show("Failed to populate table list: " + ex.Message );
            }
            finally
            {
                if (reader != null) reader.Close();
            }
        }

        private void tables_SelectedIndexChanged(object sender, System.EventArgs e)
        {
            data = new DataTable();
```

Ready | Ln 34 | Col 51 | Ch 48 | INS

MySQLTableEditor example (5)

TableEditor - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs* | Form1.cs [Design]*

TableEditor.Form1 | label1

```
            }
        }

        private void tables_SelectedIndexChanged(object sender, System.EventArgs e)
        {
            data = new DataTable();

            da = new MySqlDataAdapter("SELECT * FROM " + tables.SelectedItem.ToString()
            cb = new MySqlCommandBuilder( da );

            da.Fill( data );

            dataGrid.DataSource = data;
        }

        private void updateBtn_Click(object sender, System.EventArgs e)
        {
            DataTable changes = data.GetChanges();
            da.Update( changes );
            data.AcceptChanges();
        }
    }
}
```

Properties

Output | Error List

Ready | Ln 34 | Col 51 | Ch 48 | INS

MySQLTableEditor example (6)

Form1

Server: localhost

Connect

User Id: root

Password: ******

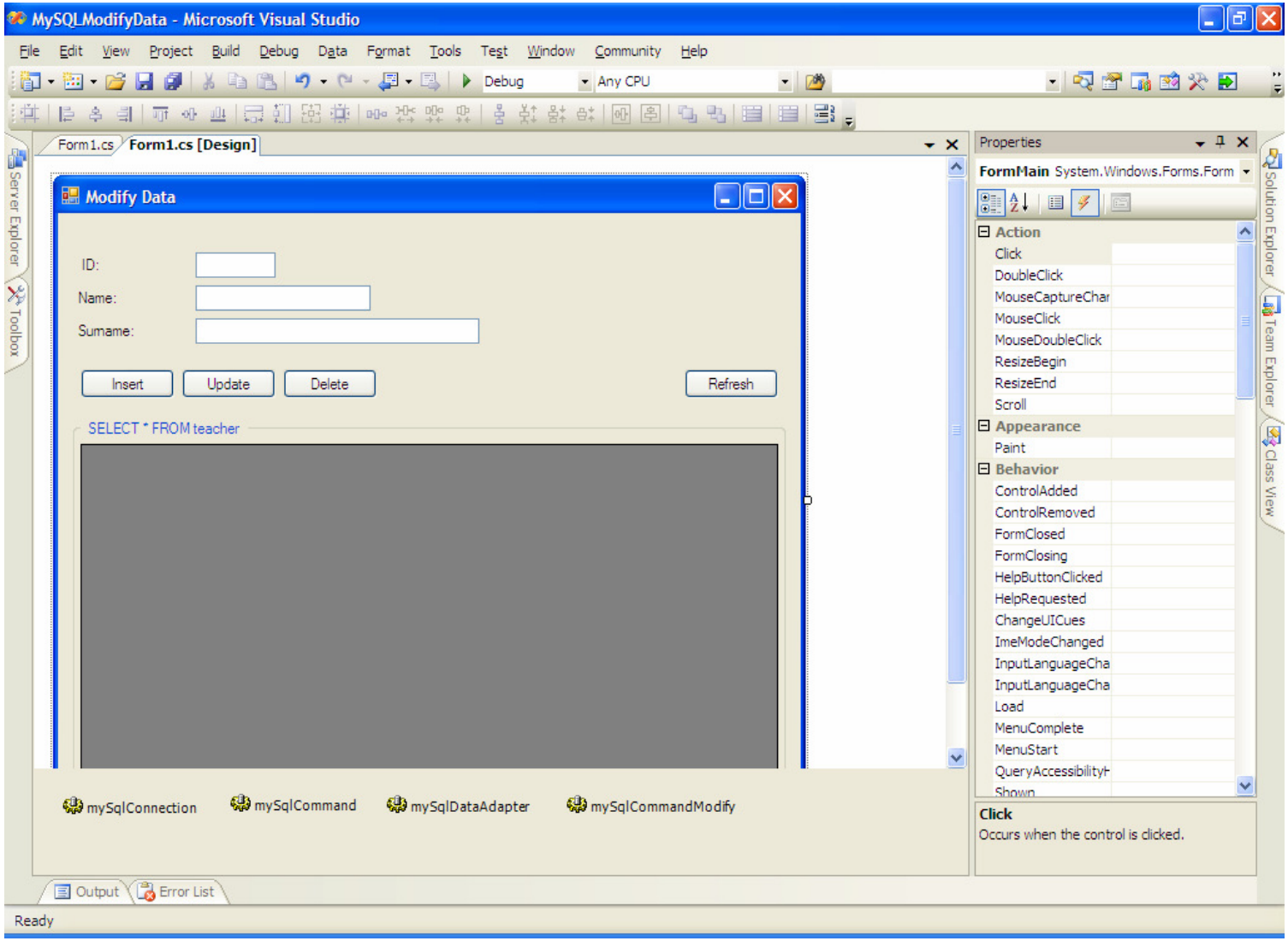
Databases test

Tables person

Update

bornnumber	name	familyname	street	city	zipcode
801106/3456	Peter	Novak	Kamenna 27	Banska Bystri	97401
800312/7845	Stanislav	Steinmüller	Zelena 9	Nove Mesto n	91501
790907/1259	Janos	Toth	Slnecne nam	Komarno	94501
810130/3695	Marek	Ratroch	Stred 49/7	Povazska By	01701
781201/1248	Bohuslav	Biely	Juh 2100/456	Trencin	91101
810514/5341	Branislav	Balaz	Tahanovce 3	Kosice	04000
781015/4431	Peter	Kapustny	Javorova 2	Zilina	01001
800407/3522	Marek	Durica	Precin 124	Precin	01701
791229/5431	Martin	Kluciar	A. Bernolaka	Zilina	01001
771124/3578	Lukas	Satrapa	Dolna 12	Cadca	02201
771203/5472	Jan	Krnac	Prievoznicka	Ruzomberok	03401
790310/2145	Juraj	Papun	Kosicka cesta	Michalovce	07101
781001/3623	Andrej	Janci	Tatranska 22	Poprad	05801
781130/4454	Zdeno	Svetkovsky	Janka Boroda	Prievidza	97101
791225/7452	Rastislav	Kontros	Kolarovce 12	Kolarovce	01401
770913/3326	Frantisek	Murgas	Namestie SN	Banska Bystri	97401

MySQLModifyData example (1)



MySQLModifyData example (2)

MySQLModifyData - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

MySQLModifyData.FormMain | buttonUpdate_Click(object sender, EventArgs e)

```
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace MySQLModifyData {
    public partial class FormMain : Form {
        private DataTable dataTable = new DataTable ();

        public FormMain () {
            InitializeComponent ();

            mySqlConnection.Open ();
            mySqlDataAdapter.Fill (dataTable);
            mySqlConnection.Close ();
            dataGridViewTeachers.DataSource = dataTable;
        }

        private void groupBoxData_Enter (object sender, EventArgs e) {

        }

        private void buttonInsert_Click (object sender, EventArgs e) {
            mySqlCommandModify.CommandText =
                "INSERT INTO teacher (teacherId, name, familyname) " +
                "VALUES ( " +
                "   '" + textBoxId.Text + "', " +
                "   '" + textBoxName.Text + "', " +
                "   '" + textBoxSurname.Text + "')";

            try {
```

Output | Error List

Ready | Ln 59 | Col 13 | Ch 4 | INS

MySQLModifyData example (3)

MySQLModifyData - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

MySQLModifyData.FormMain | buttonUpdate_Click(object sender, EventArgs e)

```
        private void buttonInsert_Click (object sender, EventArgs e) {
            mySqlCommandModify.CommandText =
                "INSERT INTO teacher (teacherId, name, familyname) " +
                "VALUES ( " +
                "    '" + textBoxId.Text + "', " +
                "    '" + textBoxName.Text + "', " +
                "    '" + textBoxSurname.Text + "')";

            try {
                mySqlConnection.Open ();
                int rowsInserted = mySqlCommandModify.ExecuteNonQuery ();
                mySqlConnection.Close ();
                MessageBox.Show ("Inserted " + rowsInserted.ToString () + " rows.");
            } catch (Exception ex) {
                MessageBox.Show (ex.Message);
            }
        }

        private void buttonRefresh_Click (object sender, EventArgs e) {
            dataTable.Clear ();
            mySqlConnection.Open ();
            mySqlDataAdapter.Fill (dataTable);
            mySqlConnection.Close ();
            dataGridViewTeachers.DataSource = dataTable;
        }

        private void dataGridViewTeachers_RowEnter (object sender, DataGridViewCellEven
            textBoxId.Text = dataGridViewTeachers.Rows[e.RowIndex].Cells[0].Value.ToStr
            textBoxName.Text = dataGridViewTeachers.Rows[e.RowIndex].Cells[1].Value.ToS
            textBoxSurname.Text = dataGridViewTeachers.Rows[e.RowIndex].Cells[2].Value.
        }
```

Output | Error List

Ready | Ln 59 | Col 13 | Ch 4 | INS

MySQLModifyData example (4)

MySQLModifyData - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

MySQLModifyData.FormMain | buttonUpdate_Click(object sender, EventArgs e)

```
            textBoxSurname.Text = dataGridViewTeachers.Rows[e.RowIndex].Cells[2].Value.
        }

        private void buttonUpdate_Click (object sender, EventArgs e) {
            mySqlCommandModify.CommandText =
                "UPDATE teacher SET " +
                "   name = '" + textBoxName.Text + "', " +
                "   familyname = '" + textBoxSurname.Text + "' " +
                "WHERE teacherId = '" + textBoxId.Text + "'";

            try {
                mySqlConnection.Open ();
                int rowsInserted = mySqlCommandModify.ExecuteNonQuery ();
                mySqlConnection.Close ();
                MessageBox.Show ("Update " + rowsInserted.ToString () + " rows.");
            } catch (Exception ex) {
                MessageBox.Show (ex.Message);
            }
        }

        private void buttonDelete_Click (object sender, EventArgs e) {
            if (MessageBox.Show (
                "Delete user " + textBoxSurname.Text + "?",
                "Delete user",
                MessageBoxButtons.YesNo,
                MessageBoxIcon.Exclamation
                ) == DialogResult.Yes)
            {
                mySqlCommandModify.CommandText =
                    "DELETE FROM teacher " +
                    "WHERE teacherId = '" + textBoxId.Text + "'";
```

Output | Error List

Ready | Ln 59 | Col 13 | Ch 4 | INS

MySQLModifyData example (5)

MySQLModifyData - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Data Tools Test Window Community Help

Debug | Any CPU

Form1.cs | Form1.cs [Design]

MySQLModifyData.FormMain | buttonUpdate_Click(object sender, EventArgs e)

```
        }

        private void buttonDelete_Click (object sender, EventArgs e) {
            if (MessageBox.Show (
                "Delete user " + textBoxSurname.Text + "?",
                "Delete user",
                MessageBoxButtons.YesNo,
                MessageBoxIcon.Exclamation
                ) == DialogResult.Yes)
            {
                mySqlCommandModify.CommandText =
                    "DELETE FROM teacher " +
                    "WHERE teacherId = '" + textBoxId.Text + "'";

                try {
                    mySqlConnection.Open ();
                    int rowsInserted = mySqlCommandModify.ExecuteNonQuery ();
                    mySqlConnection.Close ();
                    MessageBox.Show ("Update " + rowsInserted.ToString () + " rows.");
                } catch (Exception ex) {
                    MessageBox.Show (ex.Message);
                }
            }
        }
    }
}
```

Output | Error List

Ready | Ln 59 | Col 13 | Ch 4 | INS

MySQLModifyData example (5)

Modify Data

ID: KI001

Name: Stefan

Surname: Kovalik

Insert | Update | Delete | Refresh

SELECT * FROM teacher

teacherid	name	familyname	department
KI002	Peter	Varsa	KI
KDS01	Jiri	Slavik	KDS
KDS02	Petr	Cenek	KDS
KTK01	Jozef	Juricek	KTK
KMM04	Helena	Froncova	KMM
KI003	Karol	Matiasko	KI
KI005	Miroslav	Benedikovic	KI
KTK02	Peter	Gubis	KTK
KDS04	Valent	Klima	KDS
KTK03	Vladimir	Jamrich	KTK
KTK04	Anton	Kremen	KTK
KMT01	Stefan	Hitmar	KMT

Working with SQL Server in C#

Examples of code templates for:

- Connecting database - **SqlConnection**
- Executing SQL statement – **SqlCommand**
  - Using transactions
  - Using parameters
- Storing data locally – **DataSet**
  - synchronization using **SqlDataAdapter**
  - working with XML data

SqlConnection

Important methods:

- **BeginTransaction ()** returns *SqlTransaction*
- **ChangeDatabase ()** returns *void*
- **Close ()** returns *void*
- **CreateCommnad ()** returns *SqlCommand*
- **Open ()** returns *void*

Events:

- StateChange
- InfoMessage

Usage:

```
SqlConnection connection = new SqlConnection (
                        "server=localhost; database=Northwind; " +
                        "uid=coder; pwd=access"
                    );
```

or

```
string connectString = "server=localhost; database=Northwind; " +
                        "uid=coder; pwd=access "
SqlConnection connection = new SqlConnection (connectString);
```

SqlCommand

Important methods:

- **Cancel ()** returns *void*
- **CreateParameter ()** returns *SqlParameter*
- **ExecuteNonQuery ()** returns *int*
  - used with INSERT, DELETE and UPDATE statements or for DDL commands
  - returns number of affected rows
- **ExecuteReader ()** returns *SqlDataReader*
  - used with SELECT statement, for TableDirect access and stored procedures
  - additional parameter – *CloseConnection / Default / KeyInfo / SchemaOnly / SequentialAccess / SingleResult / SingleRow*
- **ExecuteScalar ()** returns *object*
- **ExecuteXmlReader ()** returns *XmlReader*
  - result is returned as an XML document
  - available only for SqlCommand (SQL Server)
- **Prepare ()** returns *void*

SqlCommand

Values of the **CommandType** property:

- **Text** – identifies that **CommandText** is SQL statement (default)
- **StoredProcedure** – **CommandText** value is the name of SP
- **TableDirect** – **CommandText** identifies the name of the table for direct access (not supported by **SqlCommand**, you must use other **Command** object).

SqlCommand

Usage - creation:

```
SqlConnection connection = new SqlConnection (connectionStrings);
SqlCommand sqlCommand = connection.CreateCommnad ();
```

or

```
SqlCommand sqlCommand = new SqlCommand ();
sqlCommand.Connection = connection;
sqlCommand.CommandText = "SELECT * FROM person ";
```

or

```
SqlCommand sqlCommand = new SqlCommand (
                                "SELECT * FROM person", connection
                                );
```

Usage – ExecuteReader ():

```
connection.Open ();
SqlDataReader sqlDataReader = sqlCommand.ExecuteReader ();
while (sqlDataReader.Read ()) {
  Console.WriteLine ("Name = " + sqlDataReader["name"]);
}
sqlDataReader.Close ();
```

SqlCommand

Usage – ExecuteScalar ():

```
sqlCommand.CommandText = "SELECT count(*) FROM Products";
connection.Open ();
int returnValue = (int) sqlCommand.ExecuteScalar ();
Console.WriteLine ("Returned value: " + returnValue);
connection.Close ();
```

Usage – ExecuteXmlReader:

```
sqlCommand.CommandText = "SELECT TOP 5 ProductID, ProductName " +
  "FROM Products ORDER BY ProductID FOR XML AUTO";
connection.Open ();
XmlReader xmlReader = sqlCommand.ExecuteXmlReader ();
xmlReader.Read ();
while (!xmlReader.EOF) {
  Console.WriteLine (xmlReader.ReadOuterXml ());
}
xmlReader.Close ();
```

SqlCommand

Usage – ExecuteNonQuery ():

```
sqlCommand.CommandText =
  "INSERT INTO Customers (CustomerID, CompanyName) " +
  "VALUES ('RUFINE', 'Are You Fine, Ltd.')";
connection.Open ();
int numberOfRows = sqlCommand.ExecuteNonQuery ();
Console.WriteLine ("Affected rows: " + numberOfRows);
connection.Close ();
```

Usage – transaction:

```
try {
        connection.Open ();
        SqlTransaction transaction = SqlConnection.BeginTransaction ();
        SqlCommand command = SqlConnection.CreateCommand ();
        command.Transaction = transaction;
        try {
                command.CommandText =
                        "DELETE FROM Orders WHERE CustomerID = 'ALFKI'";
                int rowsDeleted = command.ExecuteNonQuery ();

                command.CommandText =
                        "DELETE FROM Customers WHERE CustomerID = 'ALFKI'";
                int rowsDeleted = command.ExecuteNonQuery ();

                myTransaction.Commit ();
        } catch (Exception ex) {
                myTransaction.Rollback ();
                MessageBox.Show (ex.Message);
        }
} catch (Exception ee) {
        MessageBox.Show (ee.Message);
} finally {
        connection.Close ();
}
```

SqlCommand

Usage – parameters:

```
command.CommandText =
        "DELETE FROM Customers " +
        "WHERE CustomerID = @CustomerId";
command.Parameters.Add ("@CustomerId", SqlDbType.NChar, 5);

command.Parameters["@CustomerId"].Value = customerId;
int rowsDeleted = command.ExecuteNonQuery ()
```

Usage – stored procedures:

```
command.CommandText =
        “EXECUTE AddProduct @ProductId OUTPUT, @ProductName, @CategoryId“;
command.Parameters.Add (“@ProductId”, SqlDbType.Int);
command.Parameters[“@ProductId”].Direction = ParameterDirection.Output;
command.Parameters.Add (“@ProductName”, SqlDbType.NVarChar, 40).Value =
        “Brand New Product”;
command.Parameters.Add (“@CategoryId”, SqlDbType.Int).Value = 1;
command.ExecuteNonQuer ();
```

SqlDataAdapter

Important properties:

- **DeleteCommand** type *SqlCommand*
- **InsertCommand** type *SqlCommand*
- **SelectCommand** type *SqlCommand*
- **UpdateCommand** type *SqlCommand*

Important methods:

- **Fill ()** returns *int*
  - synchronizes rows from DataSet with database
  - returns number of synchronized rows
- **Update ()** returns *int*
  - calls appropriate command from changed rows
  - returns number of synchronized rows

SqlDataAdapter

Usage - creation:

```
SqlConnection connection = new SqlConnection (connStrings);
SqlCommand sqlCommand = connection.CreateCommnad ();
sqlCommand.CommandText = "SELECT * FROM person ";
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter ();
sqlDataAdapter.SelectCommand = sqlCommand;
```

or

```
SqlConnection connection = new SqlConnection (connStrings);
SqlCommand sqlCommand = connection.CreateCommnad ();
sqlCommand.CommandText = "SELECT * FROM person ";
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter (sqlCommand);
```

or

```
SqlConnection connection = new SqlConnection (connStrings);
string command = "SELECT * FROM person";
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter (command, connection);
```

or

```
string command = "SELECT * FROM person";
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter (command, connString);
```

DataSet

Important methods:

- **AcceptChanges ()** returns *void*
- **Clear ()** returns *void*
- **Clone ()** returns *DataSet*
- **Copy ()** returns *DataSet*
- **GetXml ()** returns *string*
- **GetXmlSchema ()** returns *string*
- **HasChanges ()** returns *bool*
- **Merge ()** returns *void*
- **ReadXml ()** returns *XmlReadMode*
- **RejectChanges ()** returns *void*
- **Reset ()** returns *void*
- **WriteXml ()** returns *void*

DataSet

Usage - creation:

```
DataSet dataSet = new DataSet ();
```

or

```
DataSet dataSet = new DataSet ("DataSet");
```

Usage - filling:

```
SqlConnection connection = new SqlConnection (connStrings);
SqlCommand sqlCommand = connection.CreateCommnad ();
sqlCommand.CommandText =
  "SELECT TOP 5 ProductId, ProductName, UnitPrice FROM Products ";
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter ();
sqlDataAdapter.SelectCommand = sqlCommand;
DataSet dataSet = new DataSet ();
int numberOfRows = sqlDataAdapter.Fill (dataSet, "Products");
connection.Close ();
```

Usage – printing data:

```
...
DataSet dataSet = new DataSet ();
int numberOfRows = sqlDataAdapter.Fill (dataSet, "Products");
connection.Close ();

DataTable dataTable = dataSet.Tables["Products"];
foreach (DataRow dataRow in dataTable.Rows) {
  Console.WriteLine ("ProductID = " + dataRow["ProductID"]);
  Console.WriteLine ("ProductName = " + dataRow["ProductName"]);
  Console.WriteLine ("UnitPrice = " + dataRow["UnitPrice"]);
}
```

or

```
foreach (DataTable dataTable in dataSet.Tables) {
  foreach (DataRow dataRow in dataTable.Rows) {
        foreach (DataColumn dataColumn in dataTable.Columns) {
            Console.WriteLine (dataColumn + " = " + dataRow[dataColumn]);
}}}
```

Fill method calls:

```
int Fill (DataSet dataSet);
int Fill (DataTable dataTable);
int Fill (DataSet dataSet, string dataTableName);
int Fill (DataSet dataSet, int startRow, int numOfRows,
          string dataTableName);
```

Usage – writing data to XML file:

```
DataSet dataSet = new DataSet ();
int numberOfRows = sqlDataAdapter.Fill (dataSet, "Products");
connection.Close ();
dataSet.WriteXml ("products.xml");
```

WriteXml method calls:

```
void WriteXml (Stream stream);
void WriteXml (string fileName);
void WriteXml (TextWriter textWriter);
void WriteXml (XmlWriter xmlWriter);
void WriteXml (Stream stream, XmlWriteMode xmlWriteMode);
void WriteXml (string fileName, XmlWriteMode xmlWriteMode);
void WriteXml (TextWriter textWriter, XmlWriteMode xmlWriteMode);
void WriteXml (XmlWriter xmlWriter, XmlWriteMode xmlWriteMode);
```

Usage – reading data from XML file:

```
dataSet.ReadXml ("products.xml");
```

ReadXml method calls:

```
void ReadXml (Stream stream);
void ReadXml (string fileName);
void ReadXml (TextReader textReader);
void ReadXml (XmlReader xmlReader);
void ReadXml (Stream stream, XmlReadMode xmlReadMode);
void ReadXml (string fileName, XmlReadMode xmlReadMode);
void ReadXml (TextReader textReader, XmlReadMode xmlReadMode);
void ReadXml (XmlReader xmlReader, XmlReadMode xmlReadMode);
```

Usage – table and column mapping:

```
SqlCommand sqlCommand = connection.CreateCommand ();
sqlCommand.CommandText = "SELECT CustomerID AS Id, CompanyName, Address " +
  "FROM Customers AS Cust WHERE CustomerID = 'ALFKI'";
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter ();
sqlDataAdapter.SelectCommand = sqlCommand;
DataSet dataSet = new DataSet ();
connection.Open ();
sqlDataAdapter.Fill (dataSet, "Customers");
connection.Close ();

DataTableMapping dataTableMapping =
  sqlDataAdapter.TableMappings.Add ("Customers", "Cust");
dataSet.Tables["Customers"].TableName = "Cust";

dataTableMapping.ColumnMappings.Add ("CustomerID", "Id");
```

XML based Web Services

Distributed technologies comparison

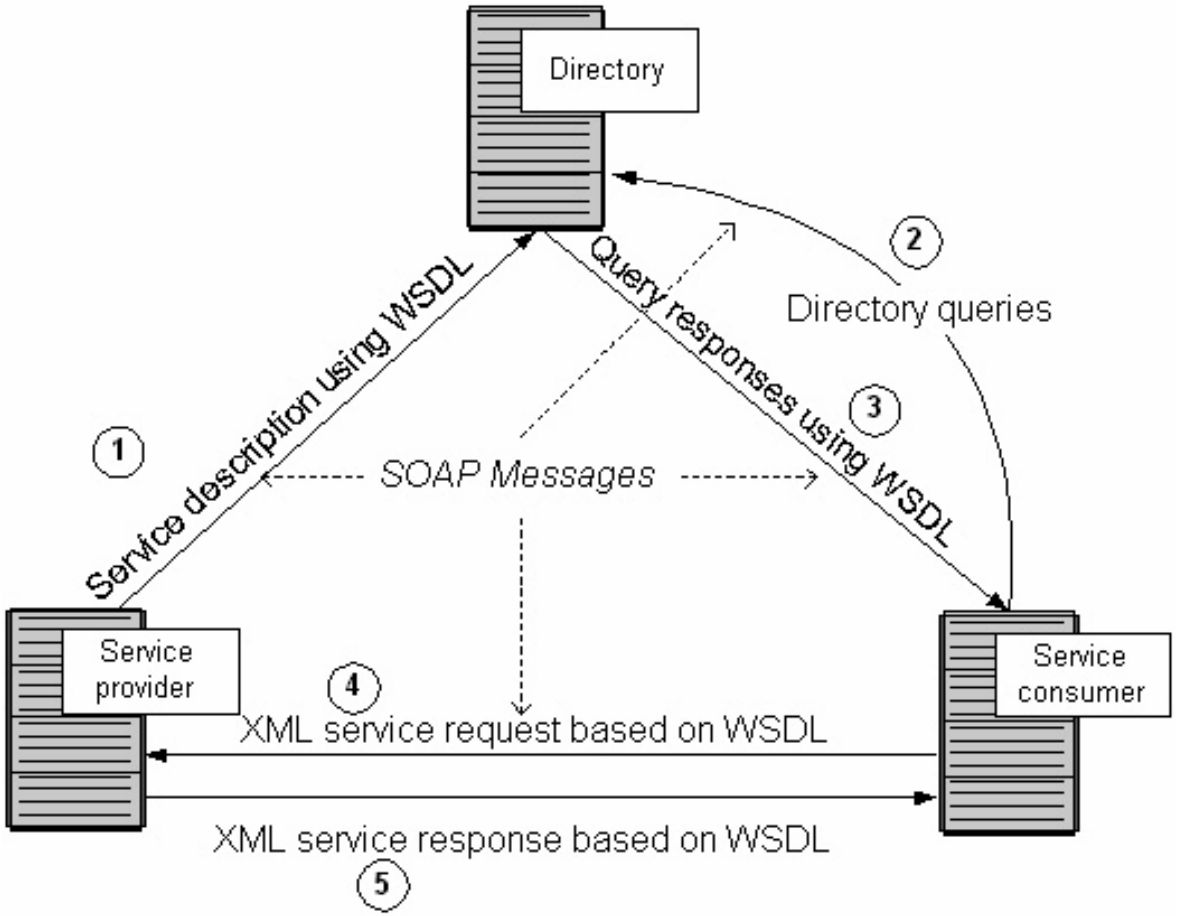
Feature	CORBA	DCOM	WS
Mechanism RPC	IIOP (Internet Inter-ORB Protocol)	DCE-RPC (Distributed Computing Environment Remote Procedure Call)	HTTP (Hypertext Transfer Protocol)
Encoding	CDR (Common Data Representation)	NDR (Network Data Representation)	XML (Extensible Markup Language)
Interface description	IDL (Interface Definition Language)	IDL (Interface Definition Language)	WSDL (Web Services Description Language)
Deployment	Služby Naming Service a Trading Service	Register	UDDI (Universal Description, Discovery and Integration)
Firewall friendly?	No	No	Yes
Protocols complexity	High	High	Low
Multiplatform?	Partially	No	Yes

WS standards

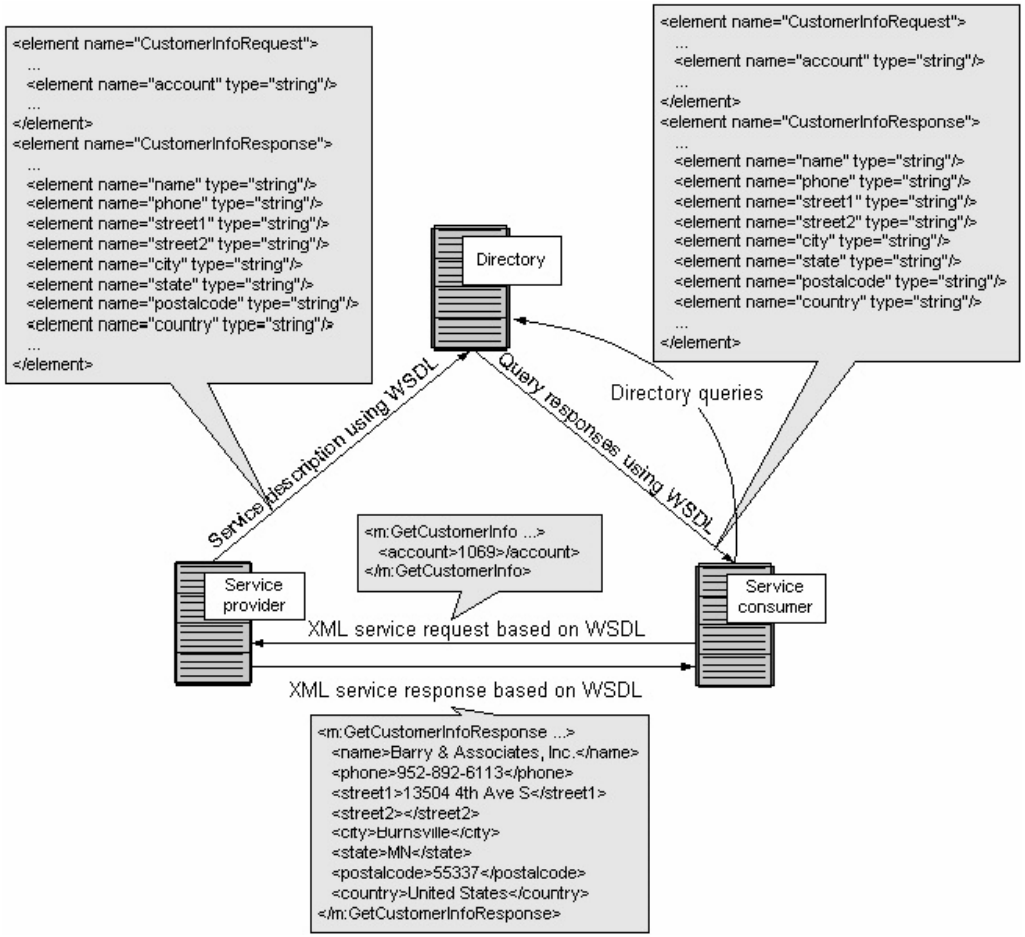
WS discovery UDDI (Universal Description, Discovery and Integration)
WS description WSDL (Web Services Description Language)
WS calling SOAP
Data encoding XML, XML Schema
Data transfer HTTP, SMTP

The key to the WS success is that they are based on open standards defined by big players such a Microsoft, IBM or Sun.

WS communication



SOAP messages



.NET technologies for WS

System.Web.Services
System.Runtime.Remoting
System.Net
System.Net.Sockets
System.Net.Sockets

WS identification

http://www.webservicex.net

WebserviceX.NET :: Web Services - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.webservicex.net/WCF/ServiceDetails.aspx?SID=51

Security Research Projects Magazines Wikipedia Vzdelávanie Záverečné práce, FRI Google IG Polárne extrémy

WebserviceX.NET :: XML Web Services... | WebserviceX.NET :: Web Services | WebserviceX.NET :: Web Services | whois Web Service | http://www.webser...is.asmx/GetWhoIS

webservicex.NET

26-Nov-2007

Home Web Services Tools Support Contact

- Business and Commerce
- Communications
- Standards and Lookup Data
- Graphics and Multimedia
- Other Web Services
- Utilities
- Value Manipulation / Unit Convertor

WHOIS

Summary

Get domain name registration record by Host Name / Domain Name (WhoIS)

Endpoint

http://www.webservicex.net/whois.asmx

Disco

http://www.webservicex.net/whois.asmx?Disco

WSDL Location

http://www.webservicex.net/whois.asmx?wsdl

Demo

whois

The following operations are supported. For a formal definition, please review the Service Description.

- GetWhoIS
  Get domain name registration record by Host Name / Domain Name (WhoIS)

http://www.webservicex.net/whois.asmx?op=GetWhoIS

WS application - whois

WindowsApplicationWS - Microsoft Visual Studio

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace WindowsApplicationWS
{
    public partial class Form1 : Form
    {
        private net.webservicex.www.whois wsWhoIs = new net.webservicex.www.whois ();


        public Form1()
        {
            InitializeComponent ();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            string result = wsWhoIs.GetWhoIS (textBox1.Text);
            richTextBox1.Text = result;
        }
    }
}
```

Bibliography

1. http://troels.arvin.dk/db/rdbms/
2. Jason Beres, *Sams Teach Yourself Visual Studio .NET 2003 in 21 Days*, Sams Publishing 2003
3. Jason Price, *Mastering C# Database Programming*, Sybex 2003, Czech translation: C# / programování databází, Grada 2005
4. Microsoft Visual C# 2005 Express Edition – Build Program Now! http://www.microsoft.com/learning/support/books/

Ing. Michal Zábovský, PhD.

michal.zabovsky@fri.uniza.sk

Department of Informatics
Faculty of Management Science and Informatics
University of Zilina

UNIVERSITY OF ZILINA
FACULTY OF MANAGEMENT SCIENCE AND INFORMATICS
DEPARTMENT OF INFORMATICS

Univerzitna 8215/1SK-01026, Zilina, Slovak Republic
Phone: +421-41-513 4181
Fax: +421-41-513 4055
Homepage: http://www.fri.uniza.sk

Introduction

The Department of Informatics comprises around 20 academics and research fellows who form research community in Computer Science. Its complement of people directly involved in research is close to 50. The Department is strongly involved in many practical collaborative industry projects and research projects on national and international level.

Research

Research addresses the fundamentals of computer systems, architectures, database systems and information analysis. The key research topics cover distributed and parallel systems and advanced database systems.