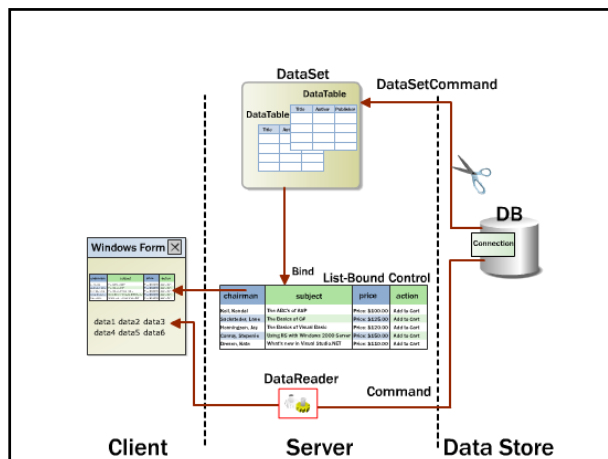


Работа с бази данни в Windows приложения

ADO.NET (*System.Data*) – класове за търсене и актуализация на бази от данни в .NET приложения.

Видове бази от данни:

- а) Microsoft SQL Server;
- б) OLE DB;
- в) Microsoft Exchange Server;
- г) XML документи.



ADO.NET Обекти

Доставчик на данни – служи като мост между приложението и източника на данни.

Предназначение:

- извлича данни от източника на данни;
- съгласува промените на данните с данните в източника.

Visual Studio 2003

Microsoft OLE DB Provider for SQL Server (SQL Server)
Microsoft JET 4.0 OLE DB Provider (MS Access)

Visual Studio 2010

.NET Framework Data Provider for SQL Server
.NET Framework Data Provider for OLE DB

Обекти Описание

Connection	Представя връзка към базата данни.
Command	Осъществява директен достъп до базата данни. Съдържа колекция Parameters , която попълва входните и изходните аргументи на SQL операторите или съхранените процедури.
DataReader	Бърз курсор, който се движи само напред сред поток от редове.
DataSet	Представя локално копие на данните от източника; има структура, подобна на реляционните бази данни; изразява йерархичния обектен модел от таблици, редове, стълбове, ограничения и релации, дефинирани за множеството от данни.
DataAdapter	Служи като мост между DataSet и източника на данни за извличане и запазване на данни.

Обект Connection

Представя връзката към базата данни.

Обект	Източник на база данни	Доставчик
SqlConnection	MS SQL Server v.7	.NET Framework Data Provider for SQL Server
OleDbConnection	OLE DB or Microsoft SQL Server	.NET Framework Data Provider for OLE DB
OdbcConnection	ODBC	.NET Framework Data Provider for ODBC
OracleConnection	Oracle	.NET Framework Data Provider for Oracle

Свойство

ConnectionString Определя връзката към определена база данни

```
SqlConnection sqlConnection1 = new SqlConnection();
```

(Visual Studio 2003)

```
sqlConnection1.ConnectionString =
    "workstation id=<име на работна станция>; packet size=4096;"
    + "integrated security=SSPI;"
    + "data source=\"<име на източника на данни>\";"
    + "persist security info=False;"
    + "initial catalog=<име на база данни>;";
```

(Visual Studio 2010)

```
sqlConnection1.ConnectionString =
    "Data Source=<име на източника на данни>;"
    + "Initial Catalog=<име на база данни>;"
    + "Integrated Security=True";
```

Обект Command

Използва SQL оператори или съхранени процедури, за да извлече данните и получените множества се връщат под формата на потоци, които могат да се

- четат от **DataReader** или
- разположат в **DataSet** обекти.

Command обект	Източник на данни
SqlCommand	MS SQL Server v.7
OleDbCommand	OLE DB or Microsoft SQL Server
OdbcCommand	ODBC
OracleCommand	Oracle

`public virtual int ExecuteNonQuery();`

Методът **ExecuteNonQuery** изпълнява SQL оператор за реализираната връзка и връща броя на резултатните редове.

Обект DataReader

Чете данни от потока.

DataReader обект	Източник на данни
SqlDataReader	MS SQL Server v.7
OleDbDataReader	OLE DB or Microsoft SQL Server
OdbcDataReader	ODBC
OracleDataReader	Oracle

`public xxxDataReader ExecuteReader ()`

Методът **Command.ExecuteReader** връща **DataReader** обект. Изпълнява **SELECT** оператор или съхранена процедура, която съдържа **SELECT** оператор. Когато приложението обработва резултатното множество с **DataReader**, връзката остава заета и когато обработката завърши, **DataReader** трябва да бъде затворен.

Обект DataAdapter

Представя локално копие на данни от източника на данни. Служи като мост между **DataSet** и източника на данни.

DataAdapter обект	Източник на данни
SqlDataAdapter	MS SQL Server v.7
OleDbDataAdapter	OLE DB or Microsoft SQL Server
OdbcDataAdapter	ODBC
OracleDataAdapter	Oracle

DataAdapter представя множество от команди и връзки, за да запълни **DataSet** и да обнови източника на данни.

Метод **Fill** – добавя/опреснява редовете в **dataSet**, като използва името на **DataSet** и създава таблица **DataTable** с име "Table"; връща броя на тези редове.

`public abstract int Fill(DataSet dataSet);`

Метод **Update** – извиква съответните оператори **INSERT**, **UPDATE** или **DELETE** за всеки добавен, обновен или изтрят ред в определеното **DataSet** от таблицата **DataTable** с име "Table"; връща броя на тези редове.

`public abstract int Update(DataSet dataSet);`

Свойства:

SelectCommand – връща данните от източника; използва се от метода **Fill**, за да избере записи от базата данни и ги разположи в **DataSet**;

InsertCommand – използва се от метода **Update**, за да вмъкне нови записи в базата данни, които съответстват на нови редове в **DataSet**.

DeleteCommand – използва се от метода **Update**, за да изтрие записи от базата данни, които съответстват на изтрети редове в **DataSet**.

UpdateCommand – използва се от метода **Update**, за да обнови записи в базата данни, които съответстват на модифицирани редове в **DataSet**.

InsertCommand, **UpdateCommand** и **DeleteCommand** се използват за промяна на данните в източника.

TableMappings – колекция за съответствието между върнатите записи и **DataSet**.

Обект DataSet

Представя данните в локална кеш памет, която функционира като релационен изглед на данните с прекъсната връзка. Не е необходима активна връзка, за да може приложението да изобрази и да обработи данните в **DataSet**. Тази архитектура използва ресурсите на базата данни само при четене от и при запис в източника на данни.

Свойства

Tables – Връща колекция от **DataTable** обекти.

Relations – Връща колекция от **DataRelation** обекти.

Методи

Clear – Изчиства данните от **DataSet** чрез премахване на всички редове във всички таблици.

Обект DataTable
 Представя таблиците в DataSet.

Свойства

Columns Връща колекция от DataColumn обекти.
Rows Връща колекция от DataRow обекти.
Constraints Връща колекция от Constraint обекти (UniqueConstraint и ForeignKeyConstraint), представящи ограничения за DataColumn обекти.
ChildRelations Връща колекция от DataRelation обекти, представящи отношение към стълб в друга таблица в DataSet (за да се създадат връзки между първични ключове и външни ключове в таблиците).

Обект DataColumn
 Представя стълбовете в DataTable.

Свойство

DataType Определя вида на данните, които съдържа всеки стълб.

Обект DataRow
 Представя редовете в DataSet.

Типизиран DataSet
 Наследява базовия клас DataSet.
 Осигурява проверка на типа по време на компилация.
 Осигурява по-бърз достъп до таблици и стълбове в DataSet чрез име вместо чрез методи, основани на колекции.

studentsDataSet1.Tables("Students");
 studentsDataSet1.Students;

Нетипизиран DataSet – достъп чрез колекция по време на изпълнение
 Типизиран DataSet – директен достъп по време на компилация

Генерира се от XML Schema (.xsd) файлове чрез използване на XSD.exe.

Обект DbCommandBuilder (Visual Studio 2005)
 Генерира автоматично команди за единствена таблица, които се използват за съгласуване на промените, направени в DataSet, със съответната таблица.

DbCommandBuilder обект	Източник на данни
SqlCommandBuilder	MS SQL Server v.7
OleDbCommandBuilder	OLE DB or Microsoft SQL Server
OdbcCommandBuilder	ODBC
OracleCommandBuilder	Oracle

Контрол DataGrid (Visual Studio 2003)
 Изобразява данните като таблица от редове и стълбове. Контролът е свързан към източник на данни с единствена таблица или с много таблици. Изобразява йерархичните отношения между таблици. Контролът осигурява потребителски интерфейс за DataSet, навигация между свързаните таблици и възможности за разширено форматиране и редкатиране.

Свойства:

DataSource Връща/установява източника (DataTable, DataSet, т.н.), съдържащ списък от стойности, използвани за попълване на елементите в контрола.
DataMember Връща/установява името на списък или таблица в източника на данни.

Метод:

```
public void SetDataBinding(object dataSource, string dataMember);
```

Установява свойствата DataSource и DataMember по време на изпълнение.

Контрол **DataGridView** (Visual Studio 2010)

Изобразява данните в потребителска таблица (замества и добавя функционалност към контрола **DataGrid**).

Свойства:

DataMember Връща/установява името на списък или таблица в източника на данни, за който контролът **DataGridView** изобразява данните.

DataSource Връща/установява източника, чиито данни се изобразяват в **DataGridView**. Предпочитаният източник на данни е компонентата **BindingSource**.

Компонента **BindingSource** (Visual Studio 2010)

Капсулира източник на данни за форма.

Свойства:

DataMember Връща/установява списък в източника на данни, към който се осъществява връзка.

DataSource Връща/установява източника на данни, към който се осъществява връзка.

Контрол **BindingNavigator** (Visual Studio 2010)

Представя навигацията и работата с графичния потребителски интерфейс на контролите на форма, които са свързани към данните.

BindingNavigator се използва с компонентата **BindingSource** за взаимодействие със записите.

Свойства:

BindingSource Връща/установява компонентата **BindingSource**, която е източник на данни.

Контрол **TableAdapter** (Visual Studio 2010)

Осигурява взаимодействие между приложението и базата данни чрез изпълнение на SQL оператори и съхранени процедури.

Създава се с **Dataset Designer** за силно типизирани **DataSet**:

- при създаване на нов **DataSet** чрез **Data Source Configuration Wizard**;
- в съществуващи **DataSet** чрез **TableAdapter Configuration Wizard**;
- чрез влачене на обекти на базата данни от **Server Explorer** в **Dataset Designer**.

Методите **TableAdapter.Insert**, **TableAdapter.Update**, и **TableAdapter.Delete** могат директно да се извикват за обработка на данните в базата данни.

Контрол **ToolStrip** (Visual Studio 2010)

Осигурява контейнер за обекти на лента с инструменти в **Windows**.

ToolStrip е контейнер за:

- **ToolStripButton**
- **ToolStripComboBox**
- **ToolStripSplitButton**
- **ToolStripLabel**
- **ToolStripSeparator**
- **ToolStripDropDownButton**
- **ToolStripProgressBar**
- **ToolStripTextBox**

Попълване на **DataSet** – Използване на **DataAdapter** за попълване на **DataSet** (VisualStudio 2003)

```
// Декларации
SqlConnection sqlConnection1;
SqlDataAdapter sqlDataAdapter1;

SqlCommand sqlSelectCommand1;
SqlCommand sqlInsertCommand1;
SqlCommand sqlDeleteCommand1;
SqlCommand sqlUpdateCommand1;

DataSet dataSet11;

DataGridView dataGrid1;
```

```
// Инициализиране на връзката
sqlConnection1 = new SqlConnection();
sqlConnection1.ConnectionString =
    "workstation id=<име на работна станция>;
    packet size=4096;
    integrated security=SSPI;
    data source=\"<име на източник на данни>\";
    persist security info=False;
    initial catalog=<име на база данни>;
```

```
// Инициализиране на адаптера на данни
sqlDataAdapter1 = new SqlDataAdapter();
sqlDataAdapter1.SelectCommand = sqlSelectCommand1;
sqlDataAdapter1.InsertCommand = sqlInsertCommand1;
sqlDataAdapter1.DeleteCommand = sqlDeleteCommand1;
sqlDataAdapter1.UpdateCommand = sqlUpdateCommand1;

sqlDataAdapter1.TableMappings.AddRange
(
    new System.Data.Common.DataTableMapping[]
    {
        new System.Data.Common.DataTableMapping
        {
            Name = "<име на таблица от източника на данни>",
            Table = "<име на таблица от DataSet>",
            ColumnMappings =
            {
                new System.Data.Common.DataColumnMapping
                {
                    Name = "<име на стълб от източника на данни>",
                    Column = "<име на стълб от DataSet>",
                },
                ...
            }
        }
    }
);
```

```
sqlSelectCommand1.CommandText = <текст на SQL SELECT оператор>;
sqlSelectCommand1.Connection = sqlConnection1;
sqlInsertCommand1.CommandText = <текст на SQL INSERT оператор>;
sqlInsertCommand1.Connection = sqlConnection1;
sqlDeleteCommand1.CommandText = <текст на SQL DELETE оператор>;
sqlDeleteCommand1.Connection = sqlConnection1;
sqlUpdateCommand1.CommandText = <текст на SQL UPDATE оператор>;
sqlUpdateCommand1.Connection = sqlConnection1;

// SELECT оператор за заявка към базата данни
sqlDataAdapter1.SelectCommand.CommandText =
    <текст на SQL SELECT оператор>;

// Използване на метод Fill за попълване на таблица в DataSet
dataSet11.Clear();
sqlDataAdapter1.Fill(dataSet11.Tables["<име на таблица>"]);

// Изобразяване на резултатния DataSet
dataGrid1.SetDataBinding(dataSet11, "<име на таблица>");
```

Попълване на DataSet – Използване на TableAdapter за попълване на DataSet (Visual Studio 2010)

```
System.ComponentModel.Container components =
    new System.ComponentModel.Container ();

DBNameDataSet dataSet = new DBNameDataSet ();

BindingSource tableNameBindingSource = new BindingSource (components);
tableNameBindingSource.DataMember = "TableName";
tableNameBindingSource.DataSource = dataSet;

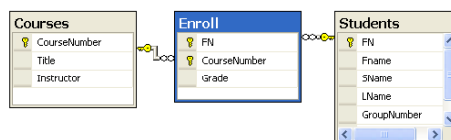
DBNameDataSetTableAdapters.TableNameTableAdapter
    tableNameTableAdapter = new
        DBNameDataSetTableAdapters.TableNameTableAdapter();
tableNameTableAdapter.ClearBeforeFill = true;

BindingNavigator tableNameBindingNavigator =
    new BindingNavigator (components);
tableNameBindingNavigator.BindingSource = tableNameBindingSource;
```

```
DataGridView grid = new DataGridView();
grid.DataSource = tableNameBindingSource;

// Използване на метод Fill за попълване на таблица в DataSet
tableNameTableAdapter.Fill(dataSet.TableName);
```

Пример: Изпълнение на проста заявка към базата данни ManagementEnroll (MS SQL Server като източник на данни), която извлича съдържанието на таблицата Courses и изобразява данните.



S3 е на таблиц
SB; 28.5.2004 г.

1. Създаване на нов Windows Application проект.
2. Свързване на приложението към базата данни ManagementEnroll
 - Data ⇒ Add New Data Source ... ⇒ Data Source Configuration Wizard
 - <L> базата данни ⇒ Next ⇒ Choose Your Data Connection

- ⇒ New Connection ... Add Connection

- Изчистване на бутона с отметка, за да се запази Connection string в приложението.
- ⇒ Next
- Избор на обектите от базата данни
- Избор на Courses от Tables ⇒ Finish
- Data ⇒ Show Data Sources
- Източници на данни
- Влачене на таблицата Courses

Появяват се отдолу DataSet, BindingSource, TableAdapter, BindingNavigator. Във формата се появяват контролът DataGridView и контролът за навигация на записите.

CourseNumber	Title	Instructor
100	Programming in C#	Mariana Goranova
150	Software Engineering	Bogdan Shishedjev
200	Information Systems	Todorika Dimitrova

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

namespace DatabaseManagementEnroll
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

```
private void coursesBindingNavigatorSaveItem_Click(object sender, EventArgs e)
{
    this.Validate();
    this.coursesBindingSource.EndEdit();
    this.coursesTableAdapter.Update
        (this.managementEnrollDataSet.Courses);
}

private void Form1_Load(object sender, EventArgs e)
{
    this.coursesTableAdapter.Fill(this.managementEnrollDataSet.Courses);
}
}
```

```
namespace DatabaseManagementEnroll
{
    partial class Form1
    {
        private System.ComponentModel.IContainer components = null;

        protected override void Dispose(bool disposing) { ... }

        #region Windows Form Designer generated code

        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.managementEnrollDataSet =
                new DatabaseManagementEnroll.ManagementEnrollDataSet();
            this.coursesBindingSource =
                new System.Windows.Forms.BindingSource(this.components);
            this.coursesTableAdapter =
                new DatabaseManagementEnroll.
                ManagementEnrollDataSetTableAdapters.CoursesTableAdapter();
            this.coursesBindingNavigator =
                new System.Windows.Forms.BindingNavigator(this.components);
        }
    }
}
```

```
this.coursesDataGridView =
    new System.Windows.Forms.DataGridView();

this.coursesBindingSource.DataMember = "Courses";
this.coursesBindingSource.DataSource =
    this.managementEnrollDataSet;

this.coursesTableAdapter.ClearBeforeFill = true;

this.coursesBindingNavigator.BindingSource =
    this.coursesBindingSource;

this.coursesDataGridView.DataSource = this.coursesBindingSource;

this.Controls.Add(this.coursesDataGridView);
this.Controls.Add(this.coursesBindingNavigator);
this.Name = "Form1";
this.Text = "Optional Subjects";
this.Load += new System.EventHandler(this.Form1_Load);
}
#endregion
```

```
private ManagementEnrollDataSet managementEnrollDataSet;
private System.Windows.Forms.BindingSource coursesBindingSource;
private DatabaseManagementEnroll.
    ManagementEnrollDataSetTableAdapters.CoursesTableAdapter
    coursesTableAdapter;
private System.Windows.Forms.BindingNavigator
    coursesBindingNavigator;
private System.Windows.Forms.DataGridView coursesDataGridView;
}
}
```

Пример: Изпълнява SQL SELECT оператори за базата данни ManagementEnroll и изобразява резултатите в контрол DataGridView.

bindingSource1 bindingNavigator1


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

namespace DatabaseQueries
{
    public partial class Form1 : Form
    {
        public Form1()
        { InitializeComponent(); }

        private void submitButton_Click(object sender, EventArgs e)
        {
            dataGridView1.DataSource = bindingSource1;
            GetData (queryTextBox.Text);
        }
    }
}
```

```
private void GetData(string selectCommand)
{
    try
    {
        // Декларация на инстанция на SqlDataAdapter с команда select
        // и connection string
        string connectionString = "Data Source=Mania;" +
            "Initial Catalog=ManagementEnroll;" +
            "Integrated Security=True";
        SqlDataAdapter adapter =
            new SqlDataAdapter(selectCommand, connectionString);

        // Декларация на инстанция на SqlCommandBuilder, за да
        // генерира автоматично команди за една таблица, използвани
        // от SqlDataAdapter
        SqlCommandBuilder commandBuilder =
            new SqlCommandBuilder(adapter);

        // Декларация на DataTable
        DataTable table = new DataTable();

        // Установяване на информация, използвана за сравнение на
        // низове в таблица
        table.Locale = System.Globalization.CultureInfo.InvariantCulture;
    }
}
```

```
// Използване на метод Fill на адаптера за попълване на
// таблицата
adapter.Fill (table);

// Установяване на таблицата като източник на данни за
// bindingSource1
bindingSource1.DataSource = table;

// Установяване на свойството AutoResizeColumns на
// dataGridView1
dataGridView1.AutoResizeColumns (
    DataGridViewAutoSizeColumnsMode.AllCellsExceptHeader);
}
catch (SqlException e)
{
    MessageBox.Show(e.Message); }
}
}
```

Предаване на параметри на оператор SELECT – колекция Parameters на обект Command

```
// SELECT оператор – SQL клиент
sqlDataAdapter1.SelectCommand.CommandText=
"SELECT * FROM TABLE WHERE (Ключ=@Ключ)";
// Установяване на стойността за обновяване – свойство Value
sqlDataAdapter1.SelectCommand.Parameters["@Key"].Value = <стойност>;
sqlDataAdapter1.Fill(dataSet11.Tables["<име на таблица>"]);

// SELECT оператор – OLE или Odbc
oleDbDataAdapter1.SelectCommand.CommandText=
"SELECT * FROM TABLE WHERE (Ключ=?);"
```

Обновяване на данни в източник на данни – метод Update

Свойствата InsertCommand, UpdateCommand и DeleteCommand на DataAdapter определят промените чрез операторите INSERT, UPDATE и DELETE. Колекцията Parameters предава информацията към обекта Command, за да определи стълба, типа на данните, размера и данните.

```
public SqlParameter Add (string parameterName,
    SqlDbType sqlDbType, int size, string sourceColumn);
```

Методът добавя параметър от тип SqlParameter към колекция с име parameterName, тип на данните sqlDbType, размер на стълба size и име на стълб в източника на данни sourceColumn.

```
// INSERT оператор – SQL клиент
sqlDataAdapter1.InsertCommand.CommandText=
"INSERT INTO <име на таблица> (<атрибут1>, <атрибут2>)
VALUES(@<атрибут1>, @<атрибут2>);"

// Добавяне на параметри чрез метод Add и установяване на техните
// стойности
sqlDataAdapter1.InsertCommand.Parameters.Add
("&@<атрибут1>", SqlDbType.<тип>, <размер>).Value = <стойност>;
sqlDataAdapter1.InsertCommand.Parameters.Add
("&@<атрибут2>", SqlDbType.<тип>, <размер>)= <стойност>;

// Обновяване на източника на данни
sqlDataAdapter1.Update(dataSet1, "<име на таблица>");
```

Пример: Попълване на DataSet и обновяване на базата данни с промени, направени в DataSet.

1. Попълване на DataSet с таблицата Enroll от базата данни ManagementEnroll чрез използване на кода, генериран от TableAdapter Query Configuration Wizard за таблицата Enroll.
2. Създаване на нов DataSet и попълване с таблицата Courses. Необходимо е да се напише целият код за попълване на DataSet с таблицата Courses.
3. Обновяване на базата данни с промените, направени в таблицата Enroll.

Обновяване на данните в базата данни чрез натискане на бутона Save.

Изпълнение на заявката чрез натискане на бутона.

Промяна на номера на дисциплината според избраното заглавие.

Избор на заглавие на дисциплина. DataGridView изобразява съответните данни от таблицата Enroll.

Потребителят променя данни.

1. Създаване на нов Windows Application проект.
2. Свързване на приложението към базата данни ManagementEnroll чрез използване на Data Source Configuration Wizard, избиране на таблицата Enroll като базов обект с данни и влачване на Enroll.

enrollBindingNavigator

courseTitleLabel

courseTitleComboBox

enrollDataGridView

3. Създаване на заявка на TableAdapter
 - <R> enrollDataAdapter => Add Query ... => Search Criteria Builder

Заявката се появява като метод FillByCourseNumber

SELECT оператор с параметър

⇒ OK

courseNumberToolStripLabel

courseNumberToolStripTextBox

fillByCourseNumberToolStripButton

fillByCourseNumberToolStrip се добавя към формата; приема входните параметри, необходими за заявката, също така и бутон за изпълнение на заявката.

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

namespace PopulatingAndUpdatingDataSets
{
    public partial class Form1 : Form
    {
        private string courseNumber;

        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

```
private void enrollBindingNavigatorSaveItem_Click(object sender,
    EventArgs e)
{
    this.Validate();
    this.enrollBindingSource.EndEdit();
    this.enrollTableAdapter.Update(this.managementEnrollDataSet.Enroll);
}

private void Form1_Load(object sender, EventArgs e)
{
    this.enrollTableAdapter.Fill(this.managementEnrollDataSet.Enroll);
    try
    {
        // Декларация на инстанция на SqlDataAdapter с команда select и
        // connection string
        string connectionString = "Data Source=Mania;" +
            "Initial Catalog=ManagementEnroll;Integrated Security=True";
        string selectCommand =
            "SELECT CourseNumber, Title FROM Courses";
        SqlDataAdapter adapter =
            new SqlDataAdapter(selectCommand, connectionString);
```

```
// Декларация на инстанция на SqlCommandBuilder, за да
// генерира автоматично команди за една таблица, използвани
// със SqlDataAdapter
SqlCommandBuilder commandBuilder =
    new SqlCommandBuilder(adapter);

// Декларация на DataTable и инициализация с низа Courses
DataTable table = new DataTable("Courses");

// Установяване на информация за сравняване на низовете в
// таблицата
table.Locale = System.Globalization.CultureInfo.InvariantCulture;

// Използване на метода Fill на адаптера за попълване на
// таблицата
adapter.Fill(table);

// Създаване на нов DataSet
DataSet dataSet = new DataSet();

// Добавяне на таблица към колекцията Tables на DataSet
dataSet.Tables.Add(table);
```

```
// Свързване на контрола courseTitleComboBox към таблицата
// Courses и установяване като източник на данни за този контрол
// таблицата Courses в dataSet
courseTitleComboBox.DataSource = dataSet.Tables["Courses"];

// Установява изобразяване на стълба Title в контрола
// courseTitleComboBox
courseTitleComboBox.DisplayMember = "Title";

// Установява съответния стълб CourseNumber на изобразявания
// стълб Title
courseTitleComboBox.ValueMember = "CourseNumber";

// Обновява номера на дисциплината според избраното заглавие
// на дисциплина
UpdateTable();

// Добавя манипулатор на събитието SelectedIndexChanged на
// контрола courseTitleComboBox
courseTitleComboBox.SelectedIndexChanged += new
    System.EventHandler(comboBox_SelectedIndexChanged);
}
catch (SqlException exp)
{
    MessageBox.Show(exp.Message);
}
}
```

```
private void UpdateTable()
{
    // Изчиства текста в courseNumberToolStripTextBox
    courseNumberToolStripTextBox.Text = "";

    // Обновява номера на дисциплината според избраното заглавие на
    // дисциплина
    courseNumber = courseTitleComboBox.SelectedValue.ToString();
}

private void comboBox_SelectedIndexChanged(object sender,
    EventArgs e)
{
    // Обновява номера на дисциплината според избраното заглавие на
    // дисциплина
    UpdateTable();

    // Установява текста в courseNumberToolStripTextBox -
    // номер на дисциплината
    courseNumberToolStripTextBox.Text = courseNumber;
}
```

```
private void fillByCourseNumberToolStripButton_Click(object sender,
    EventArgs e)
{
    try
    {
        this.enrollTableAdapter.FillByCourseNumber
            (this.managementEnrollDataSet.Enroll,
            ((int)(System.Convert.ChangeType
            (courseNumberToolStripTextBox.Text, typeof(int))));
    }
    catch (System.Exception ex)
    {
        System.Windows.Forms.MessageBox.Show(ex.Message);
    }
}
}
```

```
public virtual int FillByCourseNumber
    (ManagementEnrollDataSet.EnrollDataTable dataTable, int CourseNumber) {
    this.Adapter.SelectCommand = this.CommandCollection[1];
    this.Adapter.SelectCommand.Parameters[0].Value =
        ((int)(CourseNumber));

    if ((this.ClearBeforeFill == true)) {
        dataTable.Clear();
    }
    int returnValue = this.Adapter.Fill(dataTable);
    return returnValue;
}
```

```
namespace PopulatingAndUpdatingDataSets
{
    partial class Form1
    {
        private System.ComponentModel.IContainer components = null;

        protected override void Dispose(bool disposing) { ... }

        #region Windows Form Designer generated code

        private void InitializeComponent()
        {
            this.enrollBindingNavigator =
                new System.Windows.Forms.BindingNavigator(this.components);

            this.enrollBindingNavigatorSaveItem =
                new System.Windows.Forms.ToolStripButton();

            this.enrollDataGridView = new System.Windows.Forms.DataGridView();
            this.courseTitleLabel = new System.Windows.Forms.Label();
            this.courseTitleComboBox = new System.Windows.Forms.ComboBox();

```

```

            this.enrollBindingSource =
                new System.Windows.Forms.BindingSource(this.components);
            this.managementEnrollDataSet =
                new PopulatingAndUpdatingDataSets.ManagementEnrollDataSet();
            this.enrollTableAdapter =
                new PopulatingAndUpdatingDataSets.
                ManagementEnrollDataSetTableAdapters.EnrollTableAdapter();
            this.fillByCourseNumberToolStrip =
                new System.Windows.Forms.ToolStrip();
            this.courseNumberToolStripLabel =
                new System.Windows.Forms.ToolStripLabel();
            this.courseNumberToolStripTextBox =
                new System.Windows.Forms.ToolStripTextBox();
            this.fillByCourseNumberToolStripButton =
                new System.Windows.Forms.ToolStripButton();

            this.enrollBindingNavigator.BindingSource = this.enrollBindingSource;

            this.enrollBindingNavigatorSaveItem.Click +=
                new System.EventHandler(this.enrollBindingNavigatorSaveItem_Click);

```

```

            this.enrollDataGridView.DataSource = this.enrollBindingSource;
            this.enrollDataGridView.Dock =
                System.Windows.Forms.DockStyle.Bottom;

            this.courseTitleLabel.Text = "Choose Course Title";

            this.enrollBindingSource.DataMember = "Enroll";
            this.enrollBindingSource.DataSource = this.managementEnrollDataSet;

            this.managementEnrollDataSet.DataSetName =
                "ManagementEnrollDataSet";

            this.enrollTableAdapter.ClearBeforeFill = true;

            this.fillByCourseNumberToolStrip.Text =
                "fillByCourseNumberToolStrip";

            this.courseNumberToolStripLabel.Text = "CourseNumber.";

            this.courseNumberToolStripTextBox.Name =
                "courseNumberToolStripTextBox";

```

```

            this.fillByCourseNumberToolStripButton.Text = "FillByCourseNumber";
            this.fillByCourseNumberToolStripButton.Click += new
                System.EventHandler(this.fillByCourseNumberToolStripButton_Click);

            this.Controls.Add(this.fillByCourseNumberToolStrip);
            this.Controls.Add(this.courseTitleComboBox);
            this.Controls.Add(this.courseTitleLabel);
            this.Controls.Add(this.enrollDataGridView);
            this.Controls.Add(this.enrollBindingNavigator);
            this.Name = "Form1";
            this.Text = "Populating and Updating DataSets";
            this.Load += new System.EventHandler(this.Form1_Load);
        }
        #endregion

        private ManagementEnrollDataSet managementEnrollDataSet;
        private System.Windows.Forms.BindingSource enrollBindingSource;
        private PopulatingAndUpdatingDataSets.
            ManagementEnrollDataSetTableAdapters.EnrollTableAdapter
            enrollTableAdapter;
        private System.Windows.Forms.BindingNavigator enrollBindingNavigator;
        ...

```

```

        private System.Windows.Forms.ToolStripButton
            enrollBindingNavigatorSaveItem;
        private System.Windows.Forms.DataGridView enrollDataGridView;
        private System.Windows.Forms.Label courseTitleLabel;
        private System.Windows.Forms.ComboBox courseTitleComboBox;

        private System.Windows.Forms.ToolStrip fillByCourseNumberToolStrip;
        private System.Windows.Forms.ToolStripLabel
            courseNumberToolStripLabel;
        private System.Windows.Forms.ToolStripTextBox
            courseNumberToolStripTextBox;
        private System.Windows.Forms.ToolStripButton
            fillByCourseNumberToolStripButton;
    }
}

```

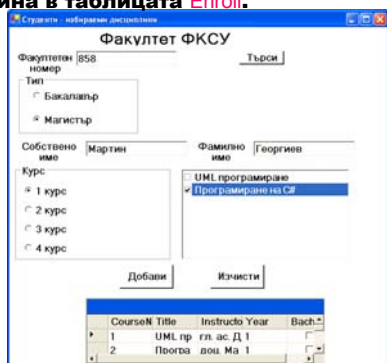
Пример: Записване на студенти в избираеми дисциплини

База данни MyDB (MS Access) – информация за студенти (Students), избираеми дисциплини (Courses) и записване на студенти в желаните дисциплини (Enroll).

```

    erDiagram
        Students ||--o{ Enroll : "1"
        Enroll ||--o{ Courses : "1"
        Students {
            string FName
            string SName
            string LName
            string Group
            string Year
            string Bachelor
        }
        Enroll {
            string FName
            string CourseNumber
            string Grade
        }
        Courses {
            string CourseNumber
            string Title
            string Instructor
            string Year
            string Bachelor
        }
    
```

При въвеждане на факултетен номер намира студент, изобразява списък от избираеми дисциплини и добавя студента с избраната дисциплина в таблицата **Enroll**.



```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using System.Data.OleDb;

namespace StudentsDB
{
    public class StudentForm : System.Windows.Forms.Form
    {
        private System.Windows.Forms.Label facultyLabel;
        private System.Windows.Forms.Label FacultyNumberLabel;
        private System.Windows.Forms.TextBox facultyNumberTextBox;
        private System.Windows.Forms.Button find;
        private System.Windows.Forms.GroupBox type;
        private System.Windows.Forms.RadioButton becheler;
        private System.Windows.Forms.RadioButton master;
        private System.Windows.Forms.Label firstNameLabel;
        private System.Windows.Forms.TextBox firstName;
        private System.Windows.Forms.Label lastNameLabel;
```

```
private System.Windows.Forms.TextBox lastName;
private System.Windows.Forms.GroupBox year;
private System.Windows.Forms.RadioButton year4;
private System.Windows.Forms.RadioButton year3;
private System.Windows.Forms.RadioButton year2;
private System.Windows.Forms.RadioButton year1;
private System.Windows.Forms.CheckedListBox courses;
private System.Windows.Forms.Button add;
private System.Windows.Forms.Button clear;
private System.Data.OleDb.OleDbConnection oleDbConnection1;
private System.Data.OleDb.OleDbDataAdapter oleDbDataAdapter1;
private System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
private System.Data.OleDb.OleDbCommand oleDbInsertCommand1;
private System.Data.OleDb.OleDbCommand oleDbUpdateCommand1;
private System.Data.OleDb.OleDbCommand oleDbDeleteCommand1;
private System.Data.DataSet dataSet1;
private System.Data.DataSet dataSet2;
private System.Data.DataSet dataSet3;
private System.Windows.Forms.DataGrid dataGrid1;
private System.ComponentModel.Container components = null;
```

```
public StudentForm()
{
    InitializeComponent();
    Reset();
}

protected override void Dispose( bool disposing )
{
    if( disposing )
    {
        if (components != null)
        {
            components.Dispose();
        }
    }
    base.Dispose( disposing );
}
```

```
#region Windows Form Designer generated code
private void InitializeComponent()
{
    // ...
    this.find.Click += new System.EventHandler(this.find_Click);
    this.add.Click += new System.EventHandler(this.add_Click);
    this.clear.Click += new System.EventHandler(this.clear_Click);
    // OleDbConnection1
    this.oleDbConnection1.ConnectionString =
        @"Jet OLEDB:Global Partial Bulk Ops=2;
        Jet OLEDB:Registry Path=;Jet OLEDB:Database Locking Mode=1;
        Data Source=""F:\Mania\C#\MyDB.mdb"";Jet OLEDB:Engine Type=5;
        Provider=""Microsoft.Jet.OLEDB.4.0"";Jet OLEDB:System database=;
        Jet OLEDB:SFP=False;persist security info=False;
        Extended Properties=;Mode=Share Deny None;
        Jet OLEDB:Encrypt Database=False;
        Jet OLEDB:Create System Database=False;
        Jet OLEDB:Don't Copy Locale on Compact=False;
        Jet OLEDB:Compact Without Replica Repair=False;
        User ID=Admin;Jet OLEDB:Global Bulk Transactions=1";
```

```
// OleDbDataAdapter1 – конфигуриране
// SELECT CourseNumber, FN, Grade FROM Enroll WHERE
// (CourseNumber = ?) AND (FN = ?)
this.oleDbDataAdapter1.DeleteCommand = this.oleDbDeleteCommand1;
this.oleDbDataAdapter1.InsertCommand = this.oleDbInsertCommand1;
this.oleDbDataAdapter1.SelectCommand = this.oleDbSelectCommand1;
this.oleDbDataAdapter1.TableMappings.AddRange(
    new System.Data.Common.DataTableMapping[] {
        new System.Data.Common.DataTableMapping("Table", "Enroll",
            new System.Data.Common.DataColumnMapping[] {
                new System.Data.Common.DataColumnMapping
                    ("CourseNumber", "CourseNumber"),
                new System.Data.Common.DataColumnMapping ("FN", "FN"),
                new System.Data.Common.DataColumnMapping
                    ("Grade", "Grade")});});
this.oleDbDataAdapter1.UpdateCommand = this.oleDbUpdateCommand1;

// StudentForm
this.Closing += new System.ComponentModel.CancelEventHandler
    (this.memberFormClosing);
#endregion
```

```
[STAThread]
static void Main()
{
    Application.Run(new StudentForm());
}

public void Reset()
{
    facultyNumberTextBox.Text="";
    firstName.Text="";
    lastName.Text="";
    becheler.Checked=true; // бакалавър
    year1.Checked=true; // 1 курс
    courses.Items.Clear();
}

private void clearClicked(object sender, System.EventArgs e)
{
    Reset();
}
```

```
private void find_Click(object sender, System.EventArgs e)
{
    int year=0;
    bool type=true; // бакалавър
    try
    {
        // Запълване на dataSet1 с данни за студент от
        // таблицата Students с търсения факултетен номер
        if (facultyNumberTextBox.Text != "")
        {
            dataSet1.Clear();
            oleDbDataAdapter1.SelectCommand.CommandText =
                "SELECT * FROM STUDENTS WHERE FN=" +
                +int.Parse(facultyNumberTextBox.Text)+",";
            oleDbDataAdapter1.Fill(dataSet1, "Students");
            dataGrid1.SetDataBinding(dataSet1, "Students");
        }
    }
}
```

```
try {
    // Извличане на данните от таблицата Students,
    // която се съдържа в dataSet1 – свойство Tables
    DataTable dataTable1=dataSet1.Tables["Students"];
    if (dataTable1.Rows.Count != 0) {
        firstName.Text=(string)dataTable1.Rows[0][1];
        lastName.Text=(string)dataTable1.Rows[0][3];
        type=(bool)dataTable1.Rows[0][6];
        becheler.Checked=type;
        master.Checked=type;
        year=(int)dataTable1.Rows[0][5]
        switch(year) {
            case 1: year1.Checked=true; break;
            case 2: year2.Checked=true; break;
            case 3: year3.Checked=true; break;
            case 4: year4.Checked=true; break;
            default:
                MessageBox.Show("Грешни данни за курс");
                break;
        }
    }
}
```

```
catch(System.Data.OleDb.OleDbException e1) {
    MessageBox.Show(e1.Message);
}

// Запълване на dataSet2 с данни за избираеми дисциплини за курса
// и типа на студента от таблицата Courses
dataSet2.Clear();
OleDbCommand cmd;
cmd = new OleDbCommand("SELECT * FROM Courses "+
    "WHERE Year = @Year AND Type = @Type", oleDbConnection1);
cmd.Parameters.Add("@Year",OleDbType.Integer).Value=year;
cmd.Parameters.Add("@Type", OleDbType.Boolean).Value=type;
oleDbDataAdapter1.SelectCommand = cmd;
oleDbDataAdapter1.Fill(dataSet2, "Courses");
```

```
courses.Items.Clear();
try
{
    // Извличане на данните от таблицата Courses,
    // която се съдържа в dataSet2 – свойство Tables
    DataTable dataTable2=dataSet2.Tables["Courses"];
    foreach (DataRow row in dataTable2.Rows) {
        // Добавяне на избираемите дисциплини в списъка
        courses.Items.Add(row[1]);
    }
}
catch(System.Data.OleDb.OleDbException e2)
{
    MessageBox.Show(e2.Message);
}
// Изобразяване на данните от таблицата Courses
dataGrid1.SetDataBinding(dataSet2, "Courses");
}
else MessageBox.Show("Неправилен факултетен номер");
}
catch(System.Data.OleDb.OleDbException ole)
{
    MessageBox.Show(ole.Message);
}
catch(InvalidOperationException ex)
{
    MessageBox.Show(ex.Message);
}
}
```

```
private void addClick(object sender, System.EventArgs e)
{
    int courseNumber;
    string courseTitle;
    int fNumber;
    try
    {
        fNumber=int.Parse(facultyNumberTextBox.Text);
        try
        {
            // Извличане на данните от таблицата Courses,
            // която се съдържа в dataSet2 – свойство Tables
            DataTable dataTable2=dataSet2.Tables["Courses"];
            foreach (DataRow row in dataTable2.Rows)
            {
                courseNumber=(int)row[0];
                courseTitle=(string)row[1];
                // Добавяне на избраните дисциплини от студента в Enroll
                System.Collections.IEnumerator en=courses.Items.GetEnumerator();
                while ( en.MoveNext() )
                {
                    int i=courses.Items.IndexOf(en.Current);
                    if((courses.GetItemChecked(i)==true) &&
                        (courses.GetItemText(en.Current).Equals(courseTitle)))
                    {
                    }
                }
            }
        }
    }
}
```

```
OleDbCommand cmd;  
// Създаване на SelectCommand за Enroll  
cmd = new OleDbCommand ("SELECT * FROM Enroll",  
    OleDbConnection1);  
oleDbDataAdapter1.SelectCommand = cmd;  
// Създаване на InsertCommand за Enroll  
cmd = new OleDbCommand  
    ("INSERT INTO Enroll (CourseNumber, FN) " +  
    "VALUES (?,?)", OleDbConnection1);  
cmd.Parameters.Add("@CourseNumber",  
    OleDbType.Integer, 4).Value=courseNumber;  
cmd.Parameters.Add("@FN", OleDbType.Integer,4).Value =  
    fNumber;  
oleDbDataAdapter1.InsertCommand = cmd;  
// Запълване на dataSet3 с таблицата Enroll  
oleDbDataAdapter1.Fill(dataSet3, "Enroll");  
// Изобразяване на dataSet3 в dataGrid1  
dataGrid1.SetDataBinding(dataSet3, "Enroll");
```

```
// Отваряне на връзката  
oleDbDataAdapter1.InsertCommand.Connection.Open();  
  
// Изпълнение на командата InsertCommand  
oleDbDataAdapter1.InsertCommand.ExecuteNonQuery();  
  
// Затваряне на връзката  
oleDbDataAdapter1.InsertCommand.Connection.Close();  
}  
}  
}  
} catch (System.Data.OleDb.OleDbException e2)  
{  
    MessageBox.Show(e2.Message);  
}  
} catch (InvalidOperationException invalidEx)  
{  
    MessageBox.Show(invalidEx.Message);  
}  
}
```

```
private void memberFormClosing(object sender,  
    System.ComponentModel.CancelEventArgs e)  
{  
    DialogResult key=MessageBox.Show  
        ("Сигурен ли сте, че искате да прекъснете?",  
        "Потвърждение", MessageBoxButtons.YesNo,  
        MessageBoxIcon.Question);  
    e.Cancel=(key==DialogResult.No);  
}  
}
```