

# VII. Windows Forms – Databases, Clipboard

## Contents

1. Databases.....	1
1.1. Connected Data Access Architecture.....	1
1.2. Disconnected Data Access Architecture .....	6
2. Clipboard.....	8

## 1. Databases

Data Access technologies:

- **ADO.NET - Active Data Objects**
- NHibernate
- Entity Framework

### Activity

1. Install DB Browser for SQLite <http://sqlitebrowser.org/>



2. Choose the option “New Database”
3. Add a new table as follows (you can also use the designer)

```
CREATE TABLE `Participant` (  
    `Id`      INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,  
    `LastName` TEXT,  
    `FirstName` TEXT,  
    `BirthDate` TEXT  
);
```

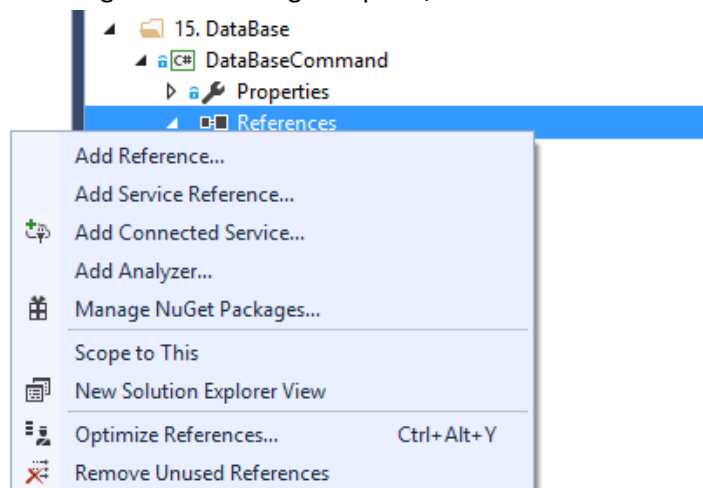
## 1.1. Connected Data Access Architecture

### Activity

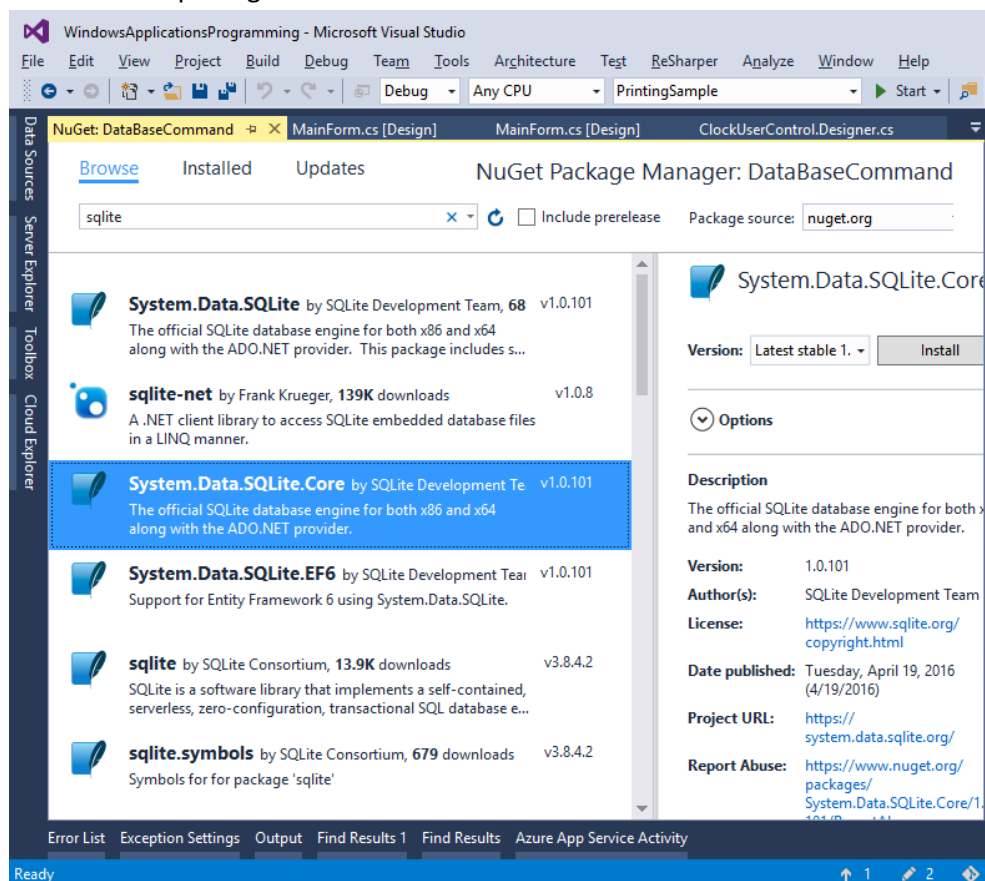
**C#** Sample code available at <http://online.ase.ro> – “DataBaseCommand” Sample

4. Create a copy of the “BasicListView” project and name it “DataBaseCommand”
5. Add SQLite libraries using NuGet (recommended) or directly from the website (<http://system.data.sqlite.org/index.html/doc/trunk/www/index.wiki>)

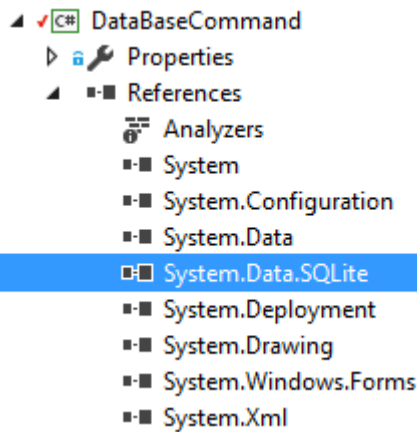
- a. Open the NuGet Package Manager by right clicking on the “References” node in the “Solution Explorer” window, and choosing the “Manage NuGet Packages” option, as shown below.



- b. Go to the “Browse” tab of the “NuGet Package Manager” and search for “sqlite”. Choose the “System.Data.SQLite.Core” package and hit the “Install” button.



- c. The package will be downloaded and Installed. A new reference to “System.Data.SQLite” will be automatically added to the “References” node, as shown below.



6. Add a new SQLiteConnection attribute (“\_dbConnection”) to the “MainForm” class as follows.

```
public partial class MainForm : Form
{
    #region Attributes
    private readonly SQLiteConnection _dbConnection;
    private readonly List<Participant> _participants;
    #endregion

    . . .
}
```

7. Instantiate the “dbConnection” attribute in the constructor of the “MainForm” class.

```
public MainForm()
{
    InitializeComponent();

    _participants = new List<Participant>();

    //Best practice
    //Define the connection string in the settings of the application and retrieve it
    using ConfigurationManager.AppSettings["ConnectionString"]
    //dbConnection = new
    SQLiteConnection(ConfigurationManager.AppSettings["ConnectionString"]);
    _dbConnection = new SQLiteConnection("Data Source=database.db");
}
```

8. Set the tag property for the ListViewItems as follows.

```
public void DisplayParticipants()
{
    lvParticipants.Items.Clear();

    foreach (Participant participant in _participants)
    {
        var listViewItem = new ListViewItem(participant.LastName);
        listViewItem.SubItems.Add(participant.FirstName);
        listViewItem.SubItems.Add(participant.BirthDate.ToShortDateString());

        listViewItem.Tag = participant;
    }
}
```

```

        lvParticipants.Items.Add(listViewItem);
    }
}

```

9. Add the method that will be used to insert new participants in the database.

```

public void AddParticipant(Participant participant)
{
    var dbCommand = new SQLiteCommand();
    dbCommand.Connection = _dbConnection;
    dbCommand.CommandText = "insert into Participant(LastName, FirstName, BirthDate)
values(@lastName,@firstName,@birthDate); SELECT last_insert_rowid()";

    try
    {
        //1. Add the new participant to the database
        _dbConnection.Open();
        dbCommand.Transaction = _dbConnection.BeginTransaction();

        var lastNameParameter = new SQLiteParameter("@lastName");
        lastNameParameter.Value = participant.LastName;
        var firstNameParameter = new SQLiteParameter("@firstName");
        firstNameParameter.Value = participant.FirstName;
        var birthDateParameter = new SQLiteParameter("@birthDate");
        birthDateParameter.Value = participant.BirthDate;

        dbCommand.Parameters.Add(lastNameParameter);
        dbCommand.Parameters.Add(firstNameParameter);
        dbCommand.Parameters.Add(birthDateParameter);

        participant.Id = (long)dbCommand.ExecuteScalar();

        dbCommand.Transaction.Commit();

        //2. Add the new participants to the local collection
        _participants.Add(participant);
    }
    catch (Exception)
    {
        dbCommand.Transaction.Rollback();
        throw;
    }
    finally
    {
        if (_dbConnection.State != ConnectionState.Closed) _dbConnection.Close();
    }
}

```

10. Change the "btnAdd\_Click" event handler as follows

```

private void btnAdd_Click(object sender, EventArgs e)
{
    var lastName = tbLastName.Text;
    var firstName = tbFirstName.Text;
    var birthDate = dtpBirthDate.Value;

    var participant = new Participant(lastName, firstName, birthDate);

    try
    {
        AddParticipant(participant);
    }
}

```

```

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

```

11. Add the method that will be used to get the existing participants from the database.

```

public void LoadParticipants()
{
    const string stringSql = "SELECT * FROM Participant";
    try
    {
        _dbConnection.Open();
        SQLiteCommand sqlCommand = new SQLiteCommand(stringSql, _dbConnection);
        SQLiteDataReader sqlReader = sqlCommand.ExecuteReader();
        try
        {
            while (sqlReader.Read())
            {
                _participants.Add(new Participant((long) sqlReader["Id"], (string)
sqlReader["LastName"], (string) sqlReader["FirstName"], DateTime.Parse((string)
sqlReader["BirthDate"])));
            }
        }
        finally
        {
            // Always call Close when done reading.
            sqlReader.Close();
        }
    }
    finally
    {
        if (_dbConnection.State != ConnectionState.Closed) _dbConnection.Close();
    }
}

```

12. Handle the Load events of the “MainForm” class as follows

```

private void MainForm_Load(object sender, EventArgs e)
{
    try
    {
        LoadParticipants();
        DisplayParticipants();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```

13. Add the method that will be used to delete existing participants from the database

```

public void DeleteParticipant(Participant participant)
{
    const string stringSql = "DELETE FROM Participant WHERE Id=@id";
}

```

```

try
{
    //Remove from the database
    _dbConnection.Open();
    SQLiteCommand sqlCommand = new SQLiteCommand(stringSql, _dbConnection);
    var idParameter = new SQLiteParameter("@id");
    idParameter.Value = participant.Id;
    sqlCommand.Parameters.Add(idParameter);

    sqlCommand.ExecuteNonQuery();

    //Remove from the local copy
    _participants.Remove(participant);
}
finally
{
    if (_dbConnection.State != ConnectionState.Closed) _dbConnection.Close();
}
}

```

14. Handle the “Delete” button as follows

```

private void btnDelete_Click(object sender, EventArgs e)
{
    if (lvParticipants.SelectedItems.Count == 0)
    {
        MessageBox.Show("Choose a participant");
        return;
    }

    if (MessageBox.Show("Are you sure?", "Delete participant", MessageBoxButtons.YesNo,
        MessageBoxIcon.Warning) ==
        DialogResult.Yes)
    {
        try
        {
            DeleteParticipant((Participant) lvParticipants.SelectedItems[0].Tag);
            DisplayParticipants();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}

```

15. Implement the edit functionality in order to allow the user to modify the data, for previously entered participants.

## 1.2. Disconnected Data Access Architecture

### Activity

**C#** Sample code available at <http://online.ase.ro> – “DataBaseDataAdapter” Sample

1. Create a copy of the “BasicListView” project and name it “DataBindingSample”.
2. Replace the “ListView” control with a “DataGrid” control (Name: dgvParticipants).
3. Modify the “MainForm” class as follows.

```

public partial class MainForm : Form
{
    private readonly SQLiteConnection _dbConnection ;
    private readonly SQLiteDataAdapter _dbDataAdapter;
    private readonly DataSet _dsParticipants;

    public MainForm()
    {
        InitializeComponent();

        //Best practice
        //Define the connection string in the settings of the application and retrieve
        it using ConfigurationManager.AppSettings["ConnectionString"]
        //var dbConnection = new
        SQLiteConnection(ConfigurationManager.AppSettings["ConnectionString"]);
        _dbConnection = new SQLiteConnection("Data Source = database.db");

        _dsParticipants = new DataSet();

        var selectCommand = new SQLiteCommand("SELECT Id, LastName, FirstName,
        BirthDate FROM Participant", _dbConnection);

        _dbDataAdapter = new SQLiteDataAdapter(selectCommand);
        _dbDataAdapter.RowUpdated += _dbDataAdapter_RowUpdated;

        var deleteCommand = new SQLiteCommand("DELETE FROM Participant WHERE Id = @Id",
        _dbConnection);
        deleteCommand.Parameters.Add(new SQLiteParameter("@Id"));
        _dbDataAdapter.DeleteCommand = deleteCommand;

        var insertCommand = new SQLiteCommand("INSERT INTO Participant (LastName,
        FirstName, BirthDate) VALUES (@LastName, @FirstName, @BirthDate);", _dbConnection);
        insertCommand.Parameters.Add(new SQLiteParameter("@LastName"));
        insertCommand.Parameters.Add(new SQLiteParameter("@FirstName"));
        insertCommand.Parameters.Add(new SQLiteParameter("@BirthDate"));
        _dbDataAdapter.InsertCommand = insertCommand;

        var updateCommand = new SQLiteCommand("UPDATE Participant SET LastName =
        @LastName, FirstName=@FirstName, BirthDate = @BirthDate WHERE Id = @Id", _dbConnection);
        updateCommand.Parameters.Add(new SQLiteParameter("@LastName", DbType.String,
        "LastName"));
        updateCommand.Parameters.Add(new SQLiteParameter("@FirstName", DbType.String,
        "LastName"));
        updateCommand.Parameters.Add(new SQLiteParameter("@BirthDate", DbType.String,
        "LastName"));
        updateCommand.Parameters.Add(new SQLiteParameter("@Id", DbType.Int64, "Id"));
        _dbDataAdapter.UpdateCommand = updateCommand;
    }

    #region Events
    private void MainForm_Load(object sender, EventArgs e)
    {
        try
        {
            _dbDataAdapter.Fill(_dsParticipants, "Participant");
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }

        //DataBinding Grid
    }
}

```

```

        dgvParticipants.DataSource = _dsParticipants.Tables["Participant"];
        //dgvParticipants.Columns["Id"].Visible = false;
    }

    private void btnAdd_Click(object sender, EventArgs e)
    {
        DataRow newParticipantRow = _dsParticipants.Tables["Participant"].NewRow();

        newParticipantRow["LastName"] = tbLastName.Text;
        newParticipantRow["FirstName"] = tbFirstName.Text;
        newParticipantRow["BirthDate"] = dtpBirthDate.Value;

        _dsParticipants.Tables["Participant"].Rows.Add(newParticipantRow);
    }

    private void btnPersistChanges_Click(object sender, EventArgs e)
    {
        try
        {
            _dbDataAdapter.Update(_dsParticipants, "Participant");
            //_dsParticipants.AcceptChanges();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }

    private void _dbDataAdapter_RowUpdated(object sender,
        System.Data.Common.RowUpdatedEventArgs e)
    {
        //https://msdn.microsoft.com/en-us/library/ks9f57t0%28v=vs.110%29.aspx
        if (e.StatementType == StatementType.Insert)
        {
            var getIdCommand = new SQLiteCommand("SELECT last_insert_rowid()",
                _dbConnection);
            e.Row["Id"] = (long)getIdCommand.ExecuteScalar();
        }
    }
    #endregion
}

```

## 2. Clipboard

### Activity

**C#** Sample code available at <http://online.ase.ro> – “ClipboardSample” Sample

1. Create a new project with the name “ClipboardSample”
2. Create the UI in Figure 1
3. Handle the Click event on the “Copy Text” button as follows

```

//Copy text from text box onto the clipboard
Clipboard.SetText(tbCopy.Text);

```

4. Handle the Click event on the “Paste Text” button as follows



```
//If clipboard has text, paste it into the text box
if (Clipboard.ContainsText())
{
    tbPaste.Text = Clipboard.GetText();
}
else
{
    MessageBox.Show("Clipboard does not contain any text");
}
```

5. Check the rest of the sample online.

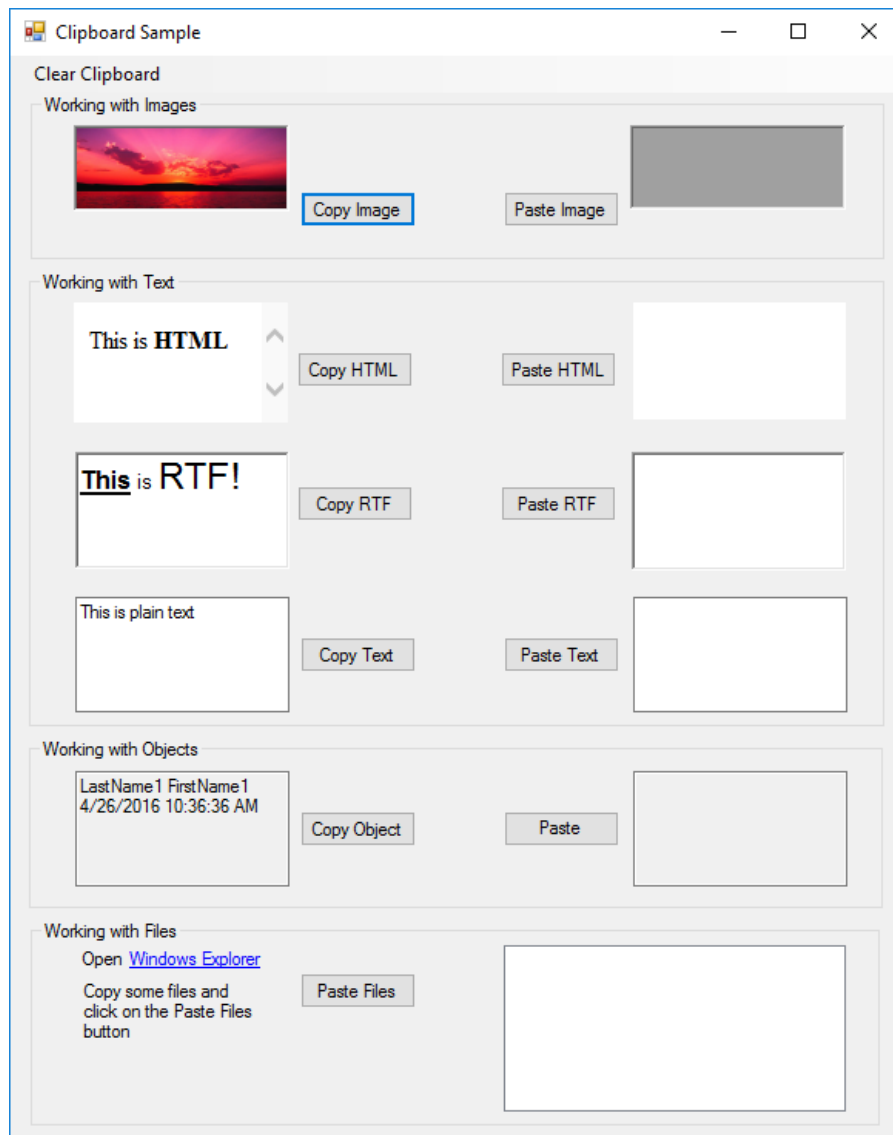


Figure 1 ClipboardSample