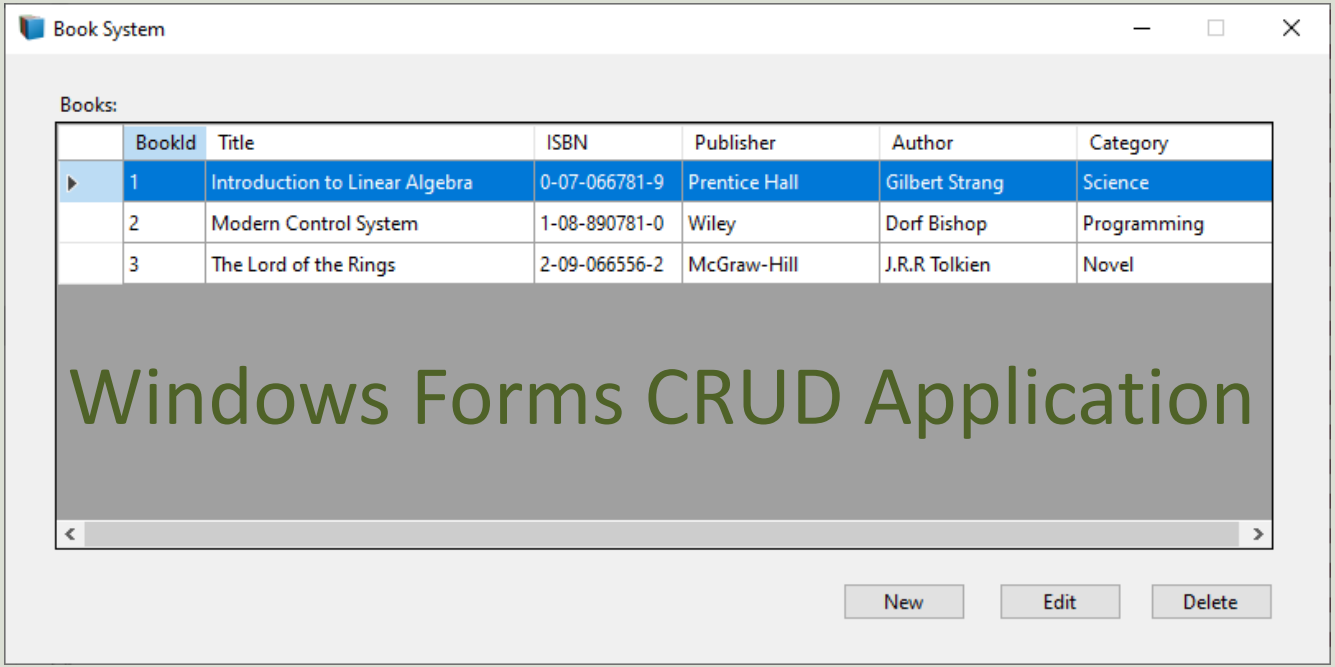
https://www.halvorsen.blog

Visual Studio, C# and SQL Server



Windows Forms CRUD Application

Hans-Petter Halvorsen

Contents

Introduction

Goal

- We will create a basic Windows Forms Application in Visual Studio
- It will communicate with an SQL Server Database
- It will Insert, Retrieve, Update and Delete Data using an SQL Server Database
- The Application will be created in iterations, i.e., step by step
- It will be some copy-paste to save time and not all details will be explained in detail
- So, it is assumed that you are familiar with basic Visual Studio and C# Programming

Application

Book System

Books:

	BookId	Title	ISBN	Publisher	Author	Category
▶	1	Introduction to Linear Algebra	0-07-066781-9	Prentice Hall	Gilbert Strang	Science
	2	Modern Control System	1-08-890781-0	Wiley	Dorf Bishop	Programming
	3	The Lord of the Rings	2-09-066556-2	McGraw-Hill	J.R.R Tolkien	Novel

New Edit Delete

NewBook

Title:

ISBN:

Publisher:

Author:

Category:

OK Cancel

EditBook

Title:
Introduction to Linear Algebra

ISBN:
0-07-066781-9

Publisher:
Prentice Hall

Author:
Gilbert Strang

Category:
Science

OK Cancel

CRUD

- **CRUD** Application means **Creating**, **Reading**, **Updating** and **Deleting** Data in a Database from your Application
- The CRUD application presented here can be a foundation for all your WinForms Applications
- Typically, all Applications today need to communicate with a Database and has CRUD functionality
- When you have learned to create a basic CRUD Application, you have all the necessary tools you need to create any kind of Application

CRUD

- **C** – **C**reate (Insert) Data into Database
- **R** – **R**ead (Select) Data from Database
- **U** – **U**pdate Data in the Database
- **D** – **D**elete Data from the Database

Application

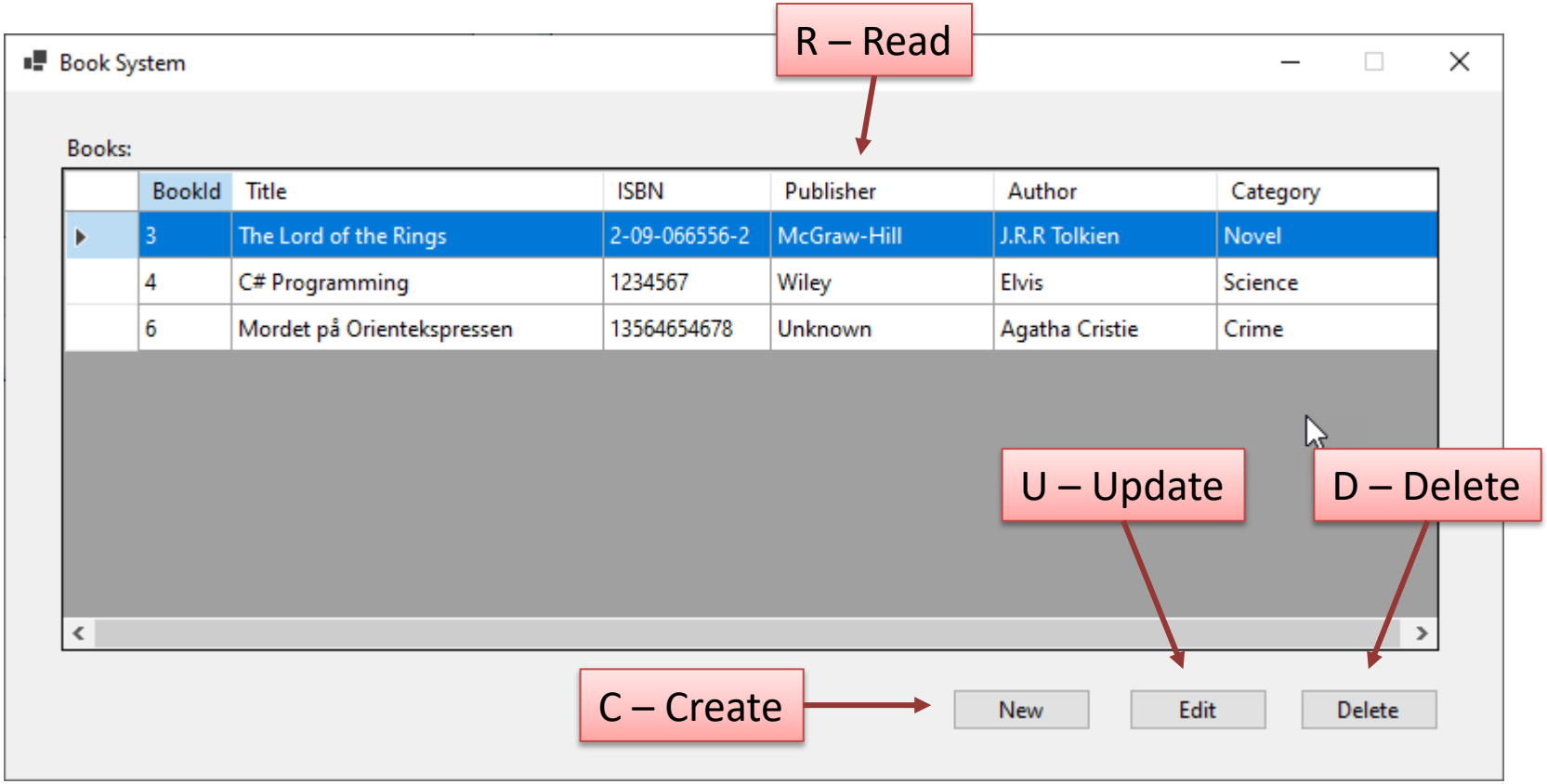
Book System

Books:

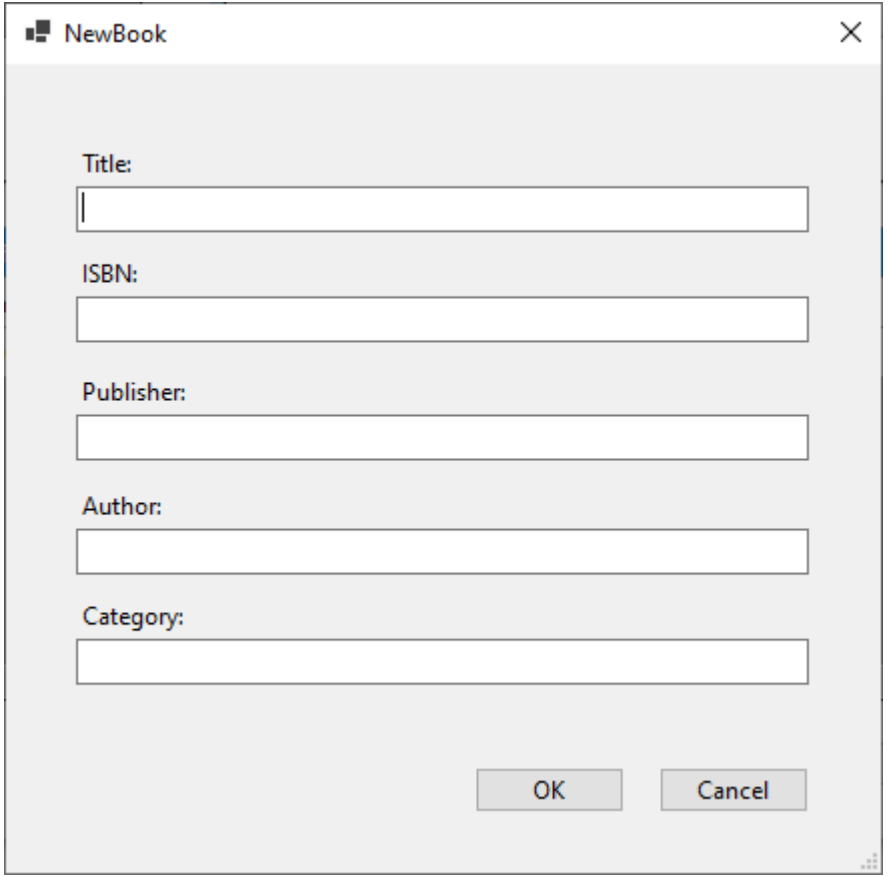
	BookId	Title	ISBN	Publisher	Author	Category
▶	3	The Lord of the Rings	2-09-066556-2	McGraw-Hill	J.R.R Tolkien	Novel
	4	C# Programming	1234567	Wiley	Elvis	Science
	6	Mordet på Orienteksprессen	13564654678	Unknown	Agatha Cristie	Crime

New Edit Delete

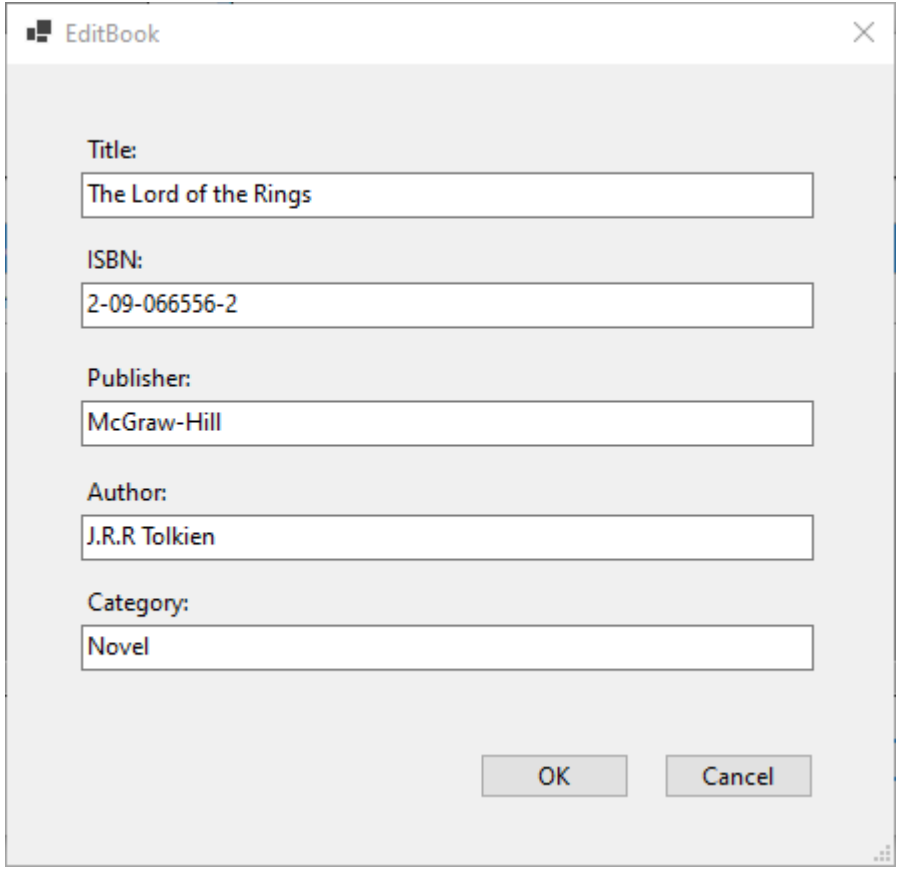
Application (CRUD)



New (C – Create Data)



Edit (U – Update Data)



We make the Application Step by Step

1. Create SQL Server Database and Tables
2. Create Windows Forms Application in Visual Studio
3. Create MainForm with DataGridView
   - Create Book Class
   - Create View for retrieving Data from Database to be shown in the DataGridView
   - Create Method for retrieving Data from Database
4. Create NewBookForm
   - Create Stored Procedure for Inserting Data into Database
   - Create Method for inserting Data into Database
5. Create EditBookForm
   - Create Stored Procedure for Updating Data in Database
   - Create Method for updating Data in Database
6. Create Delete Functionality
   - Create Stored Procedure for Deleting Data from Database
7. Finalizing the Application

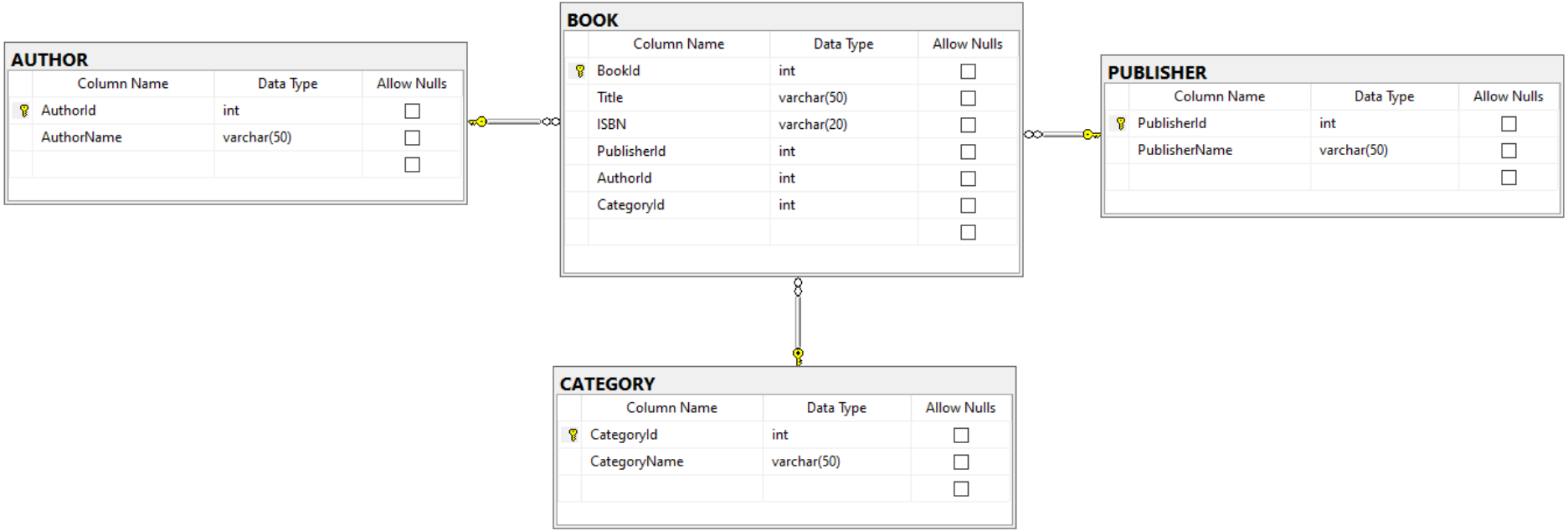
https://www.halvorsen.blog

SQL Server Database

Hans-Petter Halvorsen

Table of Contents

Tables


AUTHOR
Column Name
Data Type
Allow Nulls
AuthorId
int
AuthorName
varchar(50)
BOOK
Column Name
Data Type
Allow Nulls
BookId
int
Title
varchar(50)
ISBN
varchar(20)
PublisherId
int
AuthorId
int
CategoryId
int
PUBLISHER
Column Name
Data Type
Allow Nulls
PublisherId
int
PublisherName
varchar(50)
CATEGORY
Column Name
Data Type
Allow Nulls
CategoryId
int
CategoryName
varchar(50)

Database Table Script

```sql
CREATE TABLE [AUTHOR]
(
  [AuthorId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
  [AuthorName] [varchar](50) NOT NULL UNIQUE
)
GO
CREATE TABLE [PUBLISHER]
(
  [PublisherId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
  [PublisherName] [varchar](50) NOT NULL UNIQUE
)
GO
CREATE TABLE [CATEGORY]
(
  [CategoryId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
  [CategoryName] [varchar](50) NOT NULL UNIQUE
)
GO
CREATE TABLE [BOOK]
(
  [BookId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
  [Title] [varchar](50) NOT NULL UNIQUE,
  [ISBN] [varchar](20) NOT NULL,
  [PublisherId] [int] NOT NULL FOREIGN KEY REFERENCES [PUBLISHER] ([PublisherId]),
  [AuthorId] [int] NOT NULL FOREIGN KEY REFERENCES [AUTHOR] ([AuthorId]),
  [CategoryId] [int] NOT NULL FOREIGN KEY REFERENCES [CATEGORY] ([CategoryId])
)
GO
```

https://www.halvorsen.blog

Visual Studio

Hans-Petter Halvorsen

Table of Contents

Visual Studio Project

File Edit View Project Build Debug Format Test Analyze Tools Extensions Window Help Search (Ctrl+Q) BookSystem Sign in

Debug Any CPU BookSystem Live Share

Data Sources

Toolbox

Search Toolbox

- Microsoft.Data.SqlClient
  - Pointer
  - SqlCommand (Micr...
  - SqlCommandBuild...
  - SqlConnection (Mic...
  - SqlDataAdapter (M...
- Common Controls
  - Pointer
  - Button
  - CheckBox
  - CheckedListBox
  - ComboBox
  - DateTimePicker
  - Label
  - LinkLabel
  - ListBox
  - ListView
  - MaskedTextBox
  - MonthCalendar
  - NotifyIcon
  - NumericUpDown
  - PictureBox
  - ProgressBar
  - RadioButton
  - RichTextBox
  - TextBox
  - ToolTip
  - TreeView
- All Windows Forms
  - Pointer
  - BackgroundWorker
  - BindingSource

MainFor...Design]

Book System

Books:

New Edit Delete

Error List

Entire Solution | 0 Errors | 0 Warnings | 0 of 10 Messages | Build + IntelliSe | Search Error

Code	Description	Project	File	L...	Suppression...

Error List Output

Solution Explorer

Search Solution Explorer (Ctrl+¨)

- Solution 'BookSystem' (1 of 1 project)
  - **BookSystem**
    - Dependencies
    - Classes
      - Book.cs
    - App.config
    - Books.ico
    - EditBookForm.cs
    - MainForm.cs
    - NewBookForm.cs
    - Program.cs

Solution Explorer Team Explorer

Notifications Diagnostic Tools

Properties

MainForm System.Windows.Forms.Form

RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	**821, 409**
SizeGripStyle	Auto
StartPosition	**CenterScreen**
Tag	
Text	**Book System**
TopMost	False

Text

The text associated with the control.

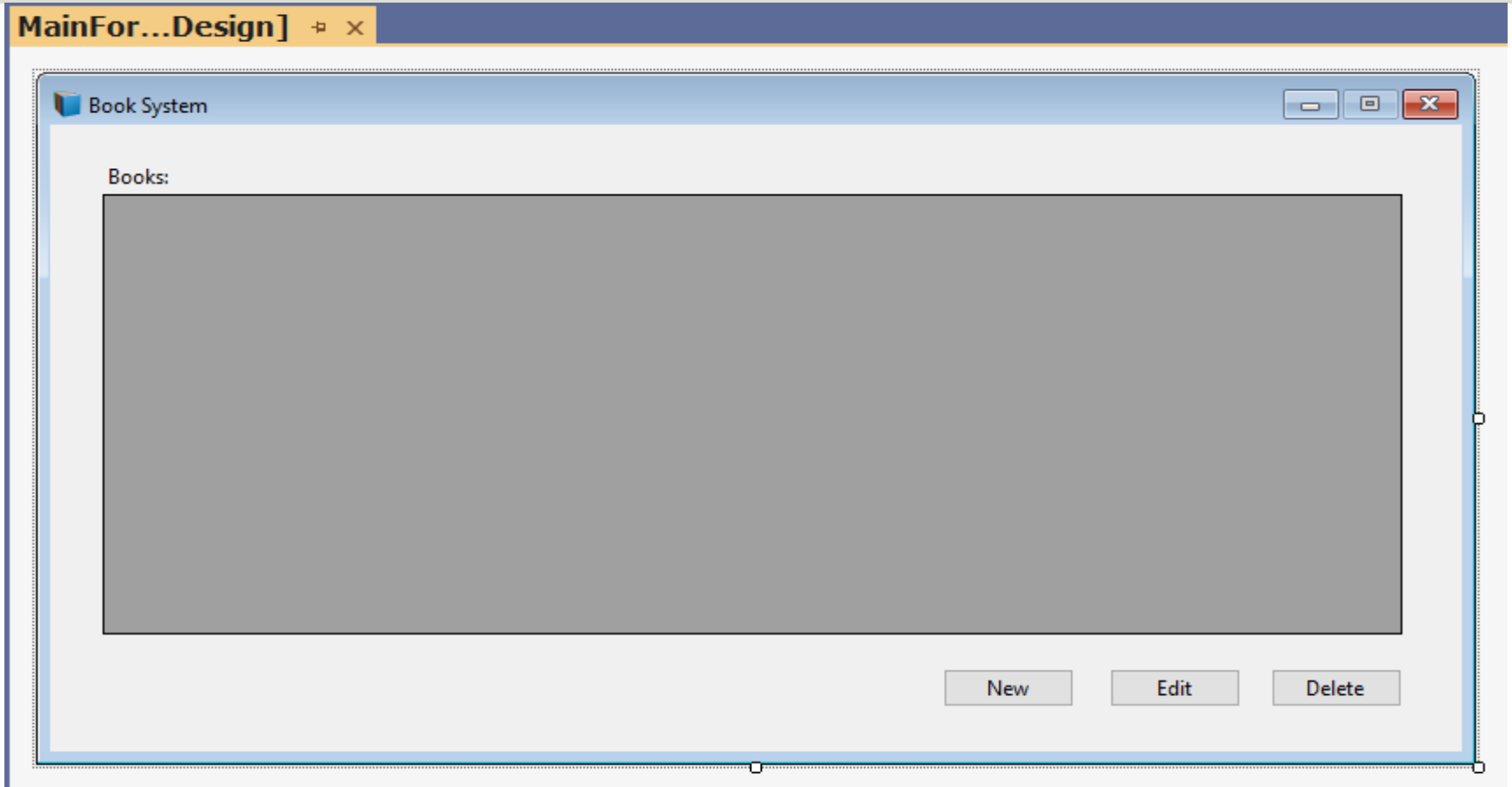
Ready 15 , 15 821 x 409 Add to Source Control

Main Form

Hans-Petter Halvorsen

Table of Contents

MainForm



What's needs to be done?

- Create a Database View for retrieving Data from the Database (“**GetBookData**”)
- Create GUI using a “**DataGridView**”
- Create a new Class (“**Book**”)
  - Add **NuGet** package: **Microsoft.Data.SqlClient**
  - Create a Method for retrieving Data from the Database (“**GetBooks**”)
- Use the “GetBooks” Method inside the Form

GetBookData Database View

```sql
CREATE VIEW GetBookData
AS

SELECT
BOOK.BookId,
BOOK.Title,
BOOK.ISBN,
PUBLISHER.PublisherName,
AUTHOR.AuthorName,
CATEGORY.CategoryName

FROM BOOK
INNER JOIN AUTHOR ON BOOK.AuthorId = AUTHOR.AuthorId
INNER JOIN PUBLISHER ON BOOK.PublisherId = PUBLISHER.PublisherId
INNER JOIN CATEGORY ON BOOK.CategoryId = CATEGORY.CategoryId

GO
```

Book Class

```
using System.Configuration;
using System.Data;
using Microsoft.Data.SqlClient;

namespace BookSystem.Classes
{
   public class Book
   {
      public int BookId { get; set; }
      public string? Title { get; set; }
      public string? Isbn { get; set; }
      public string? PublisherName { get; set; }
      public string? AuthorName { get; set; }
      public string? CategoryName { get; set; }

      string connectionString = "Data Source=XXX;Initial Catalog=XXX;Integrated Security=True;";
   }
}
```

GetBooks() Method in Book Class

```
public List<Book> GetBooks()
{
        List<Book> bookList = new List<Book>();

        SqlConnection con = new SqlConnection(connectionString);

        string selectSQL = "select BookId, Title, Isbn, PublisherName, AuthorName, CategoryName  from GetBookData";

        con.Open();

        SqlCommand cmd = new SqlCommand(selectSQL, con);

        SqlDataReader dr = cmd.ExecuteReader();

        if (dr != null)
        {
           while (dr.Read())
           {
              Book book = new Book();

              book.BookId = Convert.ToInt32(dr["BookId"]);
              book.Title = dr["Title"].ToString();
              book.Isbn = dr["ISBN"].ToString();
              book.PublisherName = dr["PublisherName"].ToString();
              book.AuthorName = dr["AuthorName"].ToString();
              book.CategoryName = dr["CategoryName"].ToString();

              bookList.Add(book);
           }
        }

        return bookList;
}
```

MainForm.cs

```
using BookSystem.Classes;
namespace BookSystem
{
    public partial class MainForm : Form
    {
        public MainForm()
        {
            InitializeComponent();
            FillGridView();
        }


        private void MainForm_FormClosed(object sender, FormClosedEventArgs e)
        {
            Application.Exit();
        }

        void FillGridView()
        {
            List<Book> bookList = new List<Book>();
            Book book = new Book();
            bookList = book.GetBooks();

            datagridviewBooks.DataSource = bookList;
        }
    }
}
```

https://www.halvorsen.blog

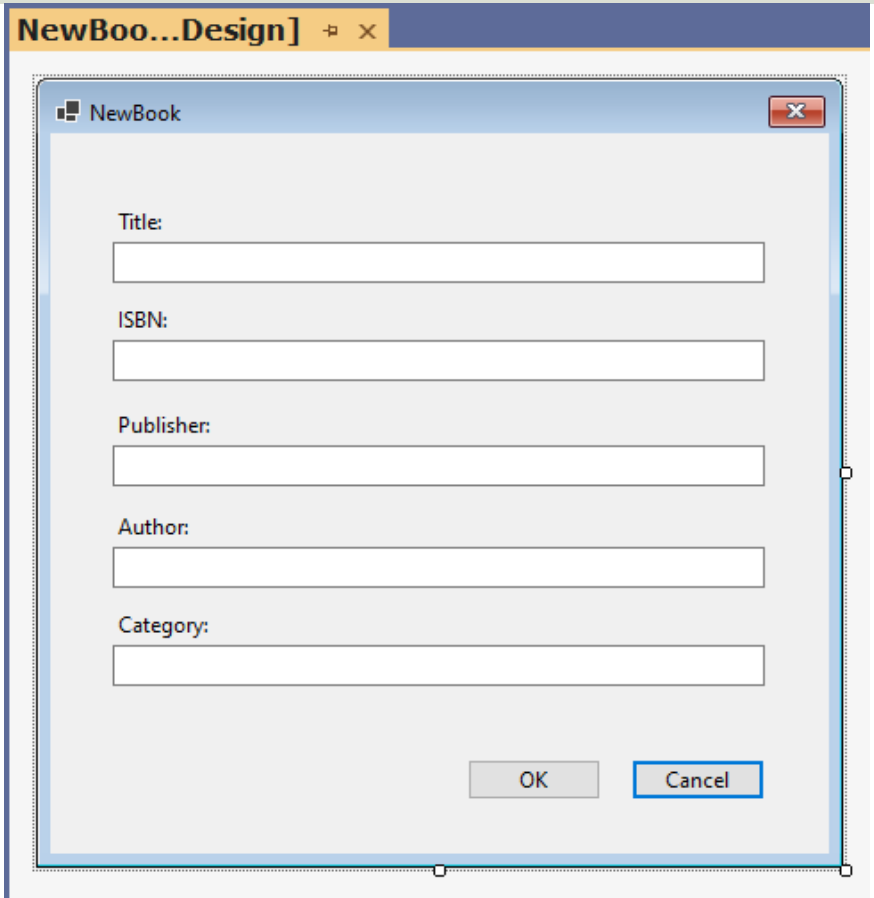
New Book

Insert New Data into the Database

Hans-Petter Halvorsen

Table of Contents

NewBookForm



What's needs to be done?

- Create a Stored Procedure for inserting Data into the Database (“**CreateBook**”)
- Create **New Button** in GUI in MainForm.cs
- Create a new **Form** (“NewBookForm.cs”)
  - Create GUI using TextBoxes, Labels and Buttons in NewBookForm.cs
- Update the “Book” Class
  - Create a Method for saving Data into the Database (“**CreateBook**”)
- Use the “CreateBook” Method inside the NewBookForm.cs

Stored Procedure **CreateBook**

```sql
CREATE PROCEDURE CreateBook
@Title varchar(50),
@Isbn varchar(20),
@PublisherName varchar(50),
@AuthorName varchar(50),
@CategoryName varchar(50)
AS

if not exists (select * from CATEGORY where CategoryName = @CategoryName)
  INSERT INTO CATEGORY (CategoryName) VALUES (@CategoryName)

if not exists (select * from AUTHOR where AuthorName = @AuthorName)
  INSERT INTO AUTHOR (AuthorName) VALUES (@AuthorName)

if not exists (select * from PUBLISHER where PublisherName = @PublisherName)
  INSERT INTO PUBLISHER (PublisherName) VALUES (@PublisherName)

if not exists (select * from BOOK where Title = @Title)
  INSERT INTO BOOK (Title, ISBN, PublisherId, AuthorId, CategoryId)
  VALUES
  (
  @Title,
  @ISBN,
  (select PublisherId from PUBLISHER where PublisherName=@PublisherName),
  (select AuthorId from AUTHOR where AuthorName=@AuthorName),
  (select CategoryId from CATEGORY where CategoryName=@CategoryName)
  )
GO
```

CreateBook() Method in Book Class

```
public void CreateBook(Book book)
{
        SqlConnection con = new SqlConnection(connectionString);
        SqlCommand cmd = new SqlCommand("CreateBook", con);
        cmd.CommandType = CommandType.StoredProcedure;

        cmd.Parameters.Add(new SqlParameter("@Title", book.Title));
        cmd.Parameters.Add(new SqlParameter("@Isbn", book.Isbn));
        cmd.Parameters.Add(new SqlParameter("@PublisherName", book.PublisherName));
        cmd.Parameters.Add(new SqlParameter("@AuthorName", book.AuthorName));
        cmd.Parameters.Add(new SqlParameter("@CategoryName", book.CategoryName));

        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();
}
```

Update MainForm.cs

```
private void btnNew_Click(object sender, EventArgs e)
{
    NewBookForm formNewBook = new NewBookForm();
    formNewBook.ShowDialog();
}
```

NewBookForm.cs

```
using BookSystem.Classes;

namespace BookSystem
{
   public partial class NewBookForm : Form
   {
      public NewBookForm()
      {
         InitializeComponent();
      }

      private void btnOK_Click(object sender, EventArgs e)
      {
         SaveBookData();
         GotoMainForm();
      }

      void SaveBookData()
      {
         Book book = new Book();

         book.Title = txtTitle.Text;
         book.Isbn = txtIsbn.Text;
         book.PublisherName = txtPublisher.Text;
         book.AuthorName = txtAuthor.Text;
         book.CategoryName = txtCategory.Text;

         book.CreateBook(book);
      }

      void GotoMainForm()
      {
         this.Close();
      }
   }
}
```

Update MainForm.cs

Make sure that the DataGridView is updated with the New Data from the Database

```
private void MainForm_Activated(object sender, EventArgs e)
{
        FillGridView();
}
```

https://www.halvorsen.blog

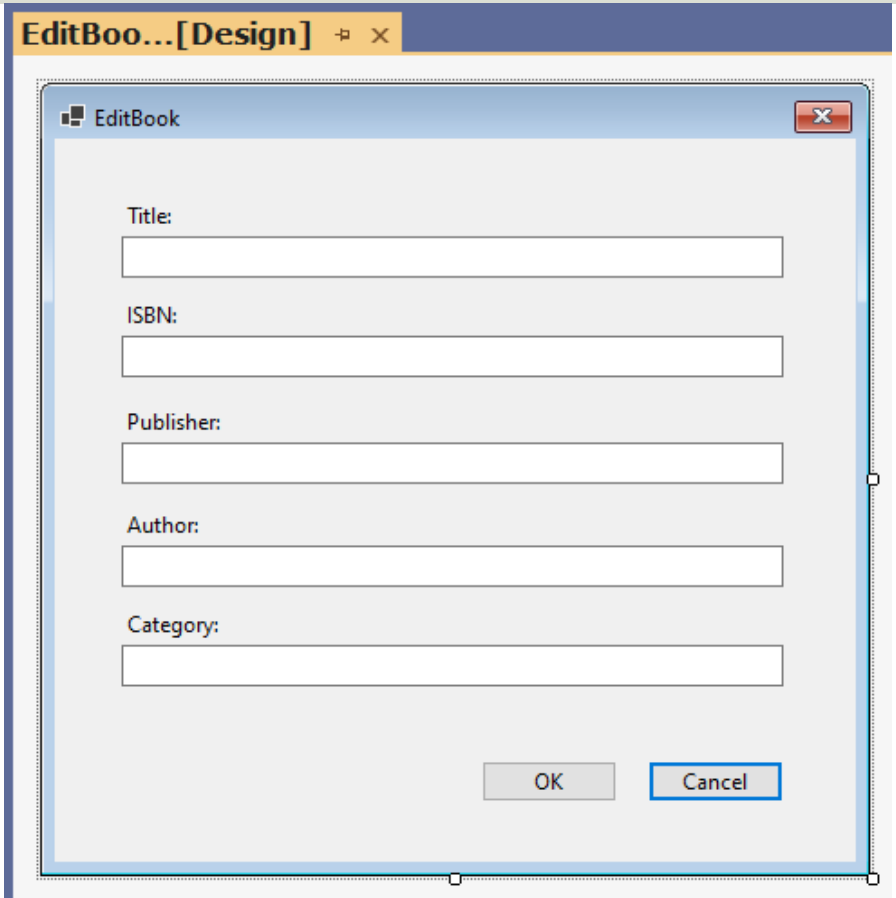
Edit Book

Update existing Data in the Database

Hans-Petter Halvorsen

Table of Contents

EditBookForm



What's needs to be done?

- Create a Stored Procedure for updating Data in the Database (“**UpdateBook**”)
- Create **Edit Button** in GUI in MainForm.cs
- Create a new **Form** (“EditBookForm.cs”)
  - Create GUI using TextBoxes, Labels and Buttons in “EditBookForm.cs”
- Update the “**Book**” Class:
  - Create a Method for retrieving Data for a selected Book from the Database (“**GetBookData**”)
  - Use the “GetBookData” Method inside the EditBookForm.cs
- Update the “Book” Class:
  - Create a Method for updating Data for a selected Book in the Database (“**EditBook**”)
  - Use the “EditBook” Method inside the EditBookForm.cs

Stored Procedure **UpdateBook**

```sql
CREATE PROCEDURE UpdateBook
@BookId int,
@Title varchar(50),
@ISBN varchar(20),
@PublisherName varchar(50),
@AuthorName varchar(50),
@CategoryName varchar(50)
AS

if not exists (select * from CATEGORY where CategoryName = @CategoryName)
  INSERT INTO CATEGORY (CategoryName) VALUES (@CategoryName)
if not exists (select * from AUTHOR where AuthorName = @AuthorName)
  INSERT INTO AUTHOR (AuthorName) VALUES (@AuthorName)
if not exists (select * from PUBLISHER where PublisherName = @PublisherName)
  INSERT INTO PUBLISHER (PublisherName) VALUES (@PublisherName)

UPDATE BOOK SET
Title = @Title,
ISBN = @ISBN,
PublisherId = (select PublisherId from PUBLISHER where PublisherName=@PublisherName),
AuthorId = (select AuthorId from AUTHOR where AuthorName=@AuthorName),
CategoryId = (select CategoryId from CATEGORY where CategoryName=@CategoryName)
WHERE BookId = @BookId
GO
```

Update MainForm.cs

```
void EditBook()
{
   int bookId;
   bookId = (int)datagridviewBooks.CurrentRow.Cells[0].Value;

   EditBookForm formEditBook = new EditBookForm(bookId);
   formEditBook.ShowDialog();
}
```

GetBookData() Method in Book Class

```
public Book GetBookData(int bookId)
{

        SqlConnection con = new SqlConnection(connectionString);

        string selectSQL = "select BookId, Title, Isbn, PublisherName, AuthorName, CategoryName from GetBookData where BookId = " + bookId;

        con.Open();

         SqlCommand cmd = new SqlCommand(selectSQL, con);

         SqlDataReader dr = cmd.ExecuteReader();

         Book book = new Book();

         if (dr != null)
         {
            while (dr.Read())
            {

                book.BookId = Convert.ToInt32(dr["BookId"]);
                book.Title = dr["Title"].ToString();
                book.Isbn = dr["ISBN"].ToString();
                book.PublisherName = dr["PublisherName"].ToString();
                book.AuthorName = dr["AuthorName"].ToString();
                book.CategoryName = dr["CategoryName"].ToString();
            }
         }

         return book;
}
```

EditBookForm.cs

```
using BookSystem.Classes;

namespace BookSystem
{
    public partial class EditBookForm : Form
    {
        int selectedBookId;

        public EditBookForm(int bookId)
        {
            InitializeComponent();
            selectedBookId = bookId;
            GetBookData();
        }

        private void btnOK_Click(object sender, EventArgs e)
        {
            GotoMainForm();
        }

        void GetBookData()
        {
            Book book = new Book();
            book = book.GetBookData(selectedBookId);

            txtTitle.Text = book.Title;
            txtIsbn.Text = book.Isbn;
            txtPublisher.Text = book.PublisherName;
            txtAuthor.Text = book.AuthorName;
            txtCategory.Text = book.CategoryName;
        }

        void GotoMainForm()
        {
            this.Close();
        }
    }
}
```

EditBook() Method in Book Class

```
public void EditBook(Book book)
{
        SqlConnection con = new SqlConnection(connectionString);
        SqlCommand cmd = new SqlCommand("UpdateBook", con);
        cmd.CommandType = CommandType.StoredProcedure;

         cmd.Parameters.Add(new SqlParameter("@BookId", book.BookId));
         cmd.Parameters.Add(new SqlParameter("@Title", book.Title));
         cmd.Parameters.Add(new SqlParameter("@Isbn", book.Isbn));
         cmd.Parameters.Add(new SqlParameter("@PublisherName", book.PublisherName));
         cmd.Parameters.Add(new SqlParameter("@AuthorName", book.AuthorName));
         cmd.Parameters.Add(new SqlParameter("@CategoryName", book.CategoryName));

         con.Open();
         cmd.ExecuteNonQuery();
}
```

Update EditBookForm.cs

```
private void btnOK_Click(object sender, EventArgs e)
{
        EditBookData();
        GotoMainForm();
}

void EditBookData()
{
        Book book = new Book();

        book.BookId = selectedBookId;
        book.Title = txtTitle.Text;
        book.Isbn = txtIsbn.Text;
        book.PublisherName = txtPublisher.Text;
        book.AuthorName = txtAuthor.Text;
        book.CategoryName = txtCategory.Text;

        book.EditBook(book);
}
```

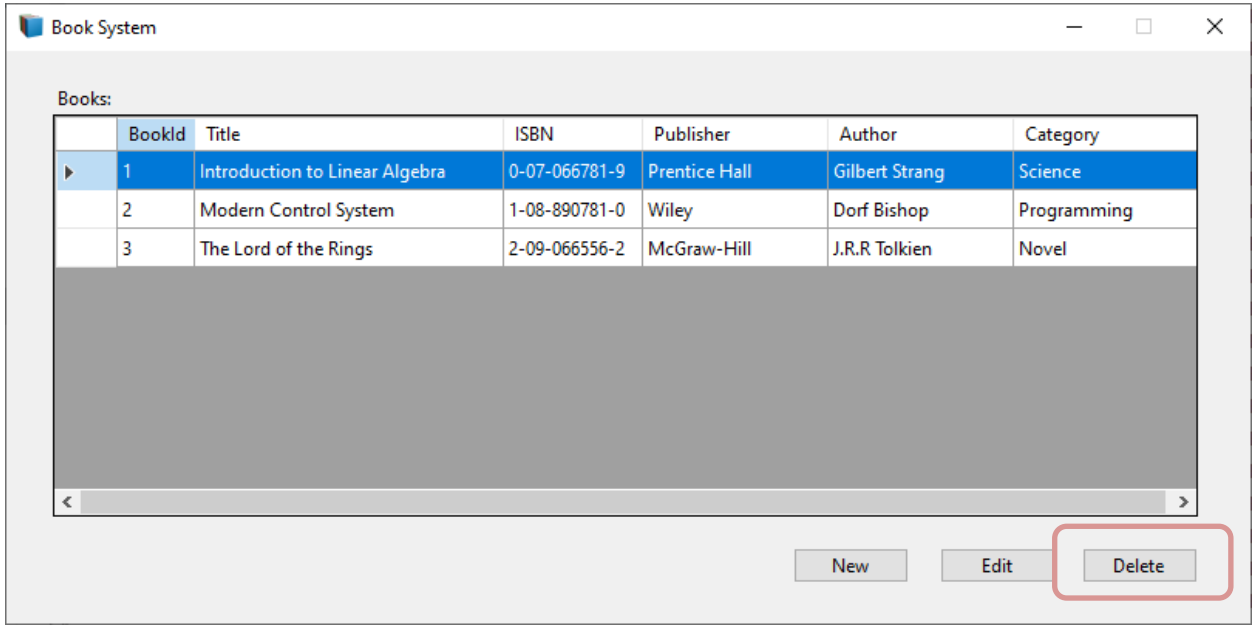
Delete Book

Delete existing Data in the Database

Hans-Petter Halvorsen

Table of Contents

Delete Book


Book System
Books:
BookId
Title
ISBN
Publisher
Author
Category
1
Introduction to Linear Algebra
0-07-066781-9
Prentice Hall
Gilbert Strang
Science
2
Modern Control System
1-08-890781-0
Wiley
Dorf Bishop
Programming
3
The Lord of the Rings
2-09-066556-2
McGraw-Hill
J.R.R Tolkien
Novel
New
Edit
Delete

What's needs to be done?

- Create a Stored Procedure for deleting Data from the Database (“**DeleteBook**”)
- Add a **Delete Button** in the MainForm. No other GUI updated needed
- Update the “Book” Class
  - Create a Method for deleting Data from the Database (“**DeleteBook**”)
- Use the “DeleteBook” Method inside the MainForm.cs

Stored Procedure DeleteBook

```
CREATE PROCEDURE DeleteBook
@BookId int
AS

delete from BOOK where BookId=@BookId

GO
```

DeleteBook Method in Book Class

```
public void DeleteBook(int bookId)
{
        SqlConnection con = new SqlConnection(connectionString);
        SqlCommand cmd = new SqlCommand("DeleteBook", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add(new SqlParameter("@BookId", bookId));

        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();
}
```

Update MainForm.cs

```
private void btnDelete_Click(object sender, EventArgs e)
{
     DeleteBook();
}

void DeleteBook()
{
     int bookId;
     bookId = (int)datagridviewBooks.CurrentRow.Cells[0].Value;

     Book book = new Book();
     book.DeleteBook(bookId);
     FillGridView();
}
```

https://www.halvorsen.blog

Finalizing the Application

Hans-Petter Halvorsen

Table of Contents

Book System

Books:

	BookId	Title	ISBN	Publisher	Author	Category
▶	1	Introduction to Linear Algebra	0-07-066781-9	Prentice Hall	Gilbert Strang	Science
	2	Modern Control System	1-08-890781-0	Wiley	Dorf Bishop	Programming
	3	The Lord of the Rings	2-09-066556-2	McGraw-Hill	J.R.R Tolkien	Novel

New Edit Delete

NewBook

Title:

ISBN:

Publisher:

Author:

Category:

OK Cancel

EditBook

Title:
Introduction to Linear Algebra

ISBN:
0-07-066781-9

Publisher:
Prentice Hall

Author:
Gilbert Strang

Category:
Science

OK Cancel

Finishing the Application

- Disable “Resize“ buttons in upper right corners. This is done in the Properties window for the Forms (**“MaximizeBox”=False**)
- NewBookForm and EditBookForm: Property: “**CancelButton=btnCancel**”
- Possible to double-click to open the “Edit Book” Form (Add “**CellDoubleClick**” Event)
- Add customized Icons for the different Forms (“Icon” Property)
- Add a **MessageBox** when clicking the Delete button – “Do you really want to delete ..”
- Change the “**Tab order**” so you can use the Tab key in order to switch between the Textboxes in the NewBook and EditBook Forms (“TabIndex” Property)
- **Adjust DataGridView**, specify Column Headers and Column Sizes
- Put Connection String into “**App.config**”. Add **NuGet** package: “System.Configuration.ConfigurationManager”
- etc.

CellDoubleClick in MainForm.cs

```
private void datagridviewBooks_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
{
  EditBook();
}
```

Update DeleteBook() in MainForm.cs

```
void DeleteBook()
{
    int bookId;
    bookId = (int)datagridviewBooks.CurrentRow.Cells[0].Value;
    string? bookTitle = datagridviewBooks.CurrentRow.Cells[1].Value.ToString();

    string message = "Are you sure that you want to delete the book '" + bookTitle + "'?";
    DialogResult dr = MessageBox.Show(message, "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    if (dr == DialogResult.Yes)
    {
        Book book = new Book();
        book.DeleteBook(bookId);
        FillGridView();
    }
}
```

Adjust DataGridView

```
public MainForm()
{
    InitializeComponent();
    FillGridView();
    AdjustGridView();
}

void AdjustGridView()
{
    datagridviewBooks.Columns[0].HeaderText = "BookId";
    datagridviewBooks.Columns[1].HeaderText = "Title";
    datagridviewBooks.Columns[2].HeaderText = "ISBN";
    datagridviewBooks.Columns[3].HeaderText = "Publisher";
    datagridviewBooks.Columns[4].HeaderText = "Author";
    datagridviewBooks.Columns[5].HeaderText = "Category";

    datagridviewBooks.Columns[0].Width = 50;
    datagridviewBooks.Columns[1].Width = 200;
    datagridviewBooks.Columns[2].Width = 90;
    datagridviewBooks.Columns[3].Width = 120;
    datagridviewBooks.Columns[4].Width = 120;
    datagridviewBooks.Columns[5].Width = 120;
}
```

Connection String in App.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
<connectionStrings>
<add name="ConnectionString"
 connectionString="Data Source=XXX;Initial Catalog=XXX;Integrated Security=True; TrustServerCertificate=True" />
</connectionStrings>
</configuration>
```

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog