7 Working with Databases in ASP.NET

Today, most of the web sites and web pages uses a database to store data that is visible on the web page, e.g. Facebook, Instagram, Twitter, etc.

7.1 Database Systems

There are lots of different database systems, or DBMS – Database Management Systems, such as:

- Microsoft SQL Server
  - Enterprise, Developer versions, etc.
  - Express version is free of charge
- Oracle
- MySQL (Oracle, previously Sun Microsystems) - MySQL can be used free of charge (open source license), Web sites that use MySQL: YouTube, Wikipedia, Facebook
- MariaDB ("The New MySQL")
- Microsoft Access
- IBM DB2
- Sybase
- … lots of other systems

7.1.1 Mirosoft SQL Server

Microsoft is the vendor of SQL Server. We have different editions of SQL Server, where SQL Server Express is free to download and use. We will focus on SQL Server in this tutorial.

7.2 SQL

SQL (Structured Query Language) is a database computer language designed for managing data in relational database management systems (RDBMS).

SQL, is a standardized computer language that was originally developed by IBM for querying, altering and defining relational databases, using declarative statements.

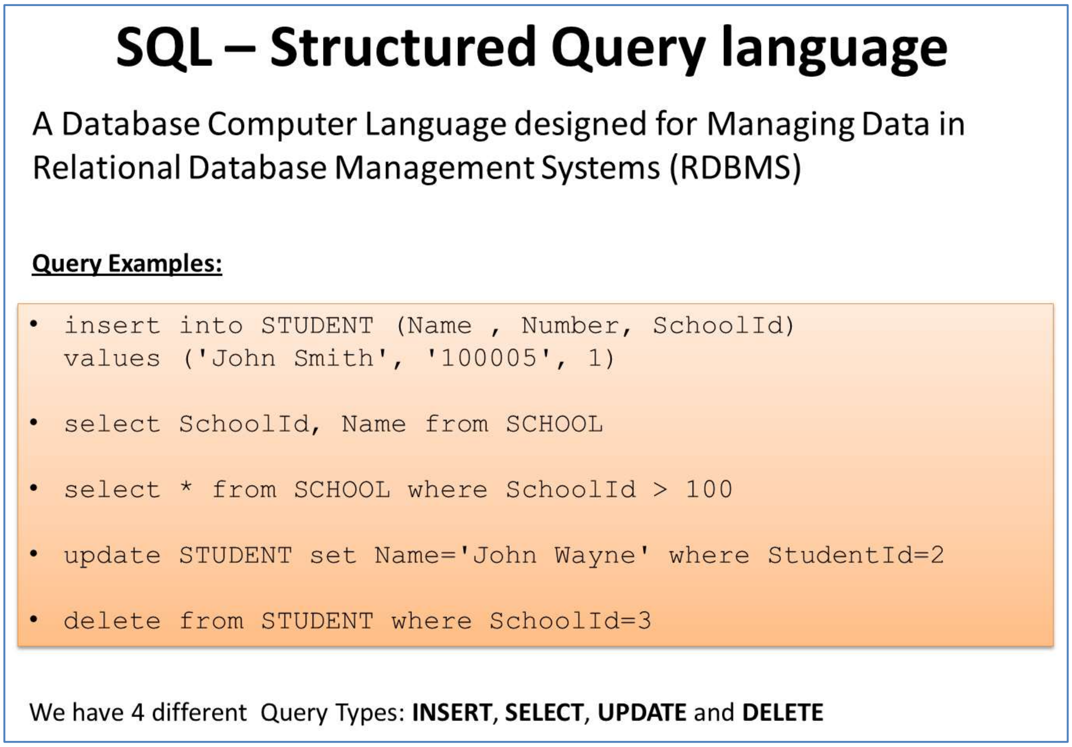
SQL is pronounced /ˌɛs kjuː 'ɛl/ (letter by letter) or /'siːkwəl/ (as a word).

What can SQL do?

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

Even if SQL is a standard, many of the database systems that exist today implement their own version of the SQL language. In this document we will use the Microsoft SQL Server as an example.

Here are some examples of SQL queries:



SQL – Structured Query language

A Database Computer Language designed for Managing Data in Relational Database Management Systems (RDBMS)

Query Examples:

```
• insert into STUDENT (Name , Number, SchoolId)
  values ('John Smith', '100005', 1)

• select SchoolId, Name from SCHOOL

• select * from SCHOOL where SchoolId > 100

• update STUDENT set Name='John Wayne' where StudentId=2

• delete from STUDENT where SchoolId=3
```

We have 4 different Query Types: **INSERT**, **SELECT**, **UPDATE** and **DELETE**

In this Tutorial we will focus on Microsoft SQL Server. SQL Server uses T-SQL (Transact-SQL). T-SQL is Microsoft's proprietary extension to SQL. T-SQL is very similar to standard SQL, but in addition it supports some extra functionality, built-in functions, etc.

In order to use databases in our applications we need to know **Structured Query language (SQL)**. For more information about SQL, see the following Tutorial: Structured Query Language (SQL).

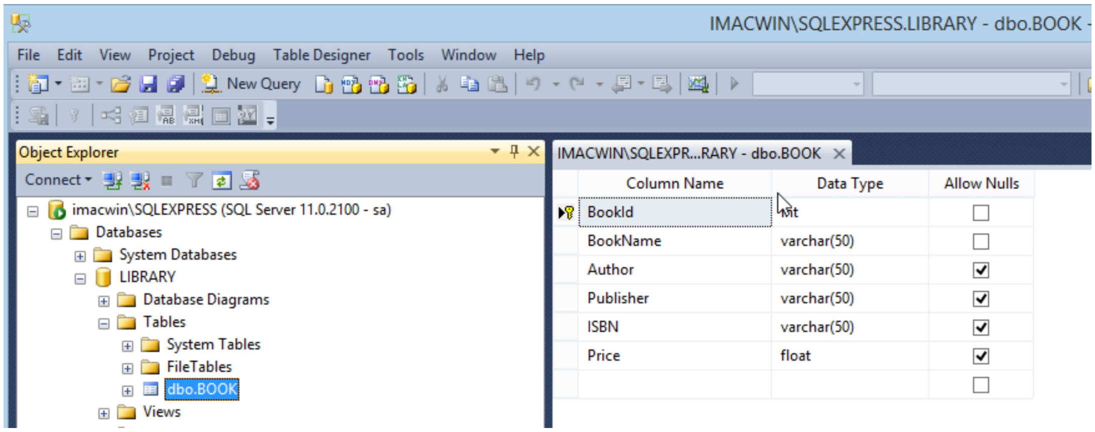
The Tutorials are available from: https://www.halvorsen.blog

7.3 SQL Server + ASP.NET

SQL Server + ASP.NET = Powerful Web Applications

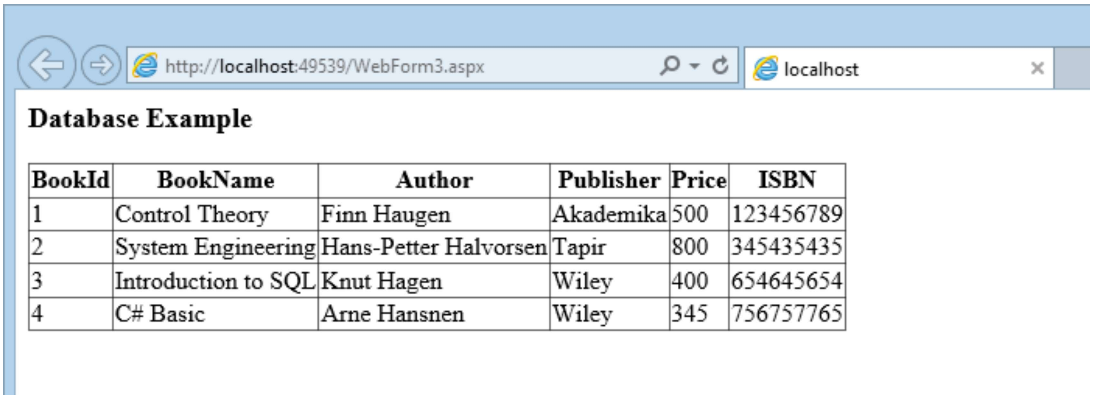
7.3.1 Simple Database Examplee

Create a Database called LIBRARY and a BOOK table in SQL Server. Enter some Test data into the BOOK table.


IMACWIN\SQLEXPRESS.LIBRARY - dbo.BOOK -
File Edit View Project Debug Table Designer Tools Window Help
New Query
Object Explorer
Connect
imacwin\SQLEXPRESS (SQL Server 11.0.2100 - sa)
Databases
System Databases
LIBRARY
Database Diagrams
Tables
System Tables
FileTables
dbo.BOOK
Views
IMACWIN\SQLEXPR...RARY - dbo.BOOK

Column Name	Data Type	Allow Nulls
BookId	int	□
BookName	varchar(50)	□
Author	varchar(50)	☑
Publisher	varchar(50)	☑
ISBN	varchar(50)	☑
Price	float	☑
		□

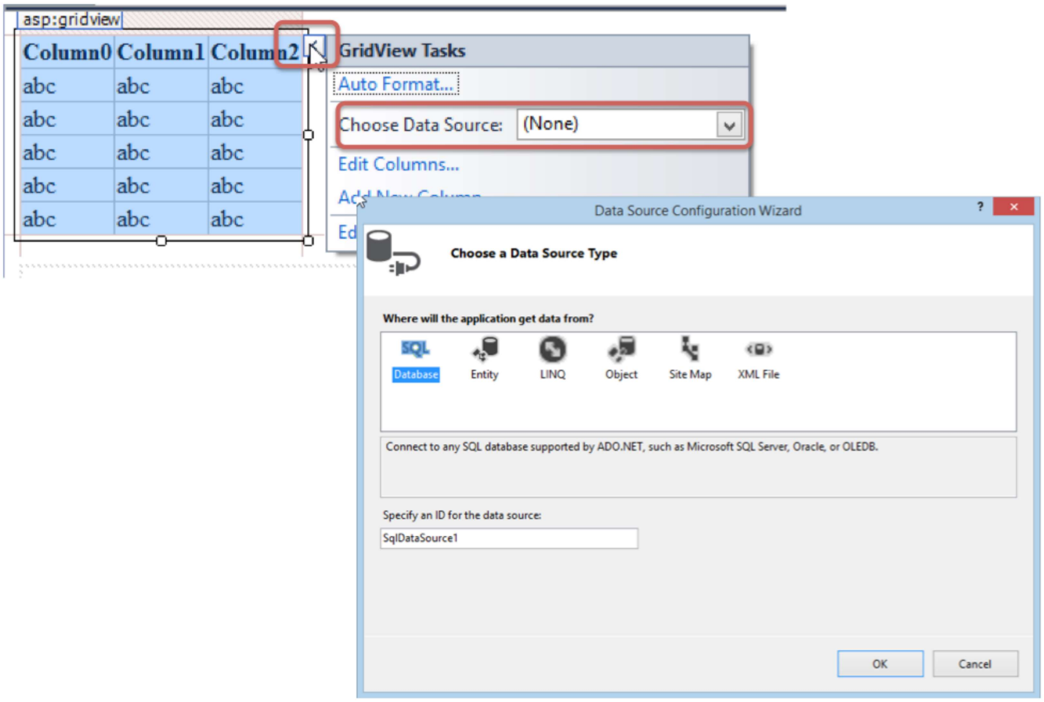
Fill a "GridView" with data from the Database Table (BOOK), as shown below



Database Example

BookId	BookName	Author	Publisher	Price	ISBN
1	Control Theory	Finn Haugen	Akademika	500	123456789
2	System Engineering	Hans-Petter Halvorsen	Tapir	800	345435435
3	Introduction to SQL	Knut Hagen	Wiley	400	654645654
4	C# Basic	Arne Hansnen	Wiley	345	756757765

Use the “Wizard” in order to create all the “magic” you need in your application:



Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

Choose Data Source: (None)

Choose a Data Source Type

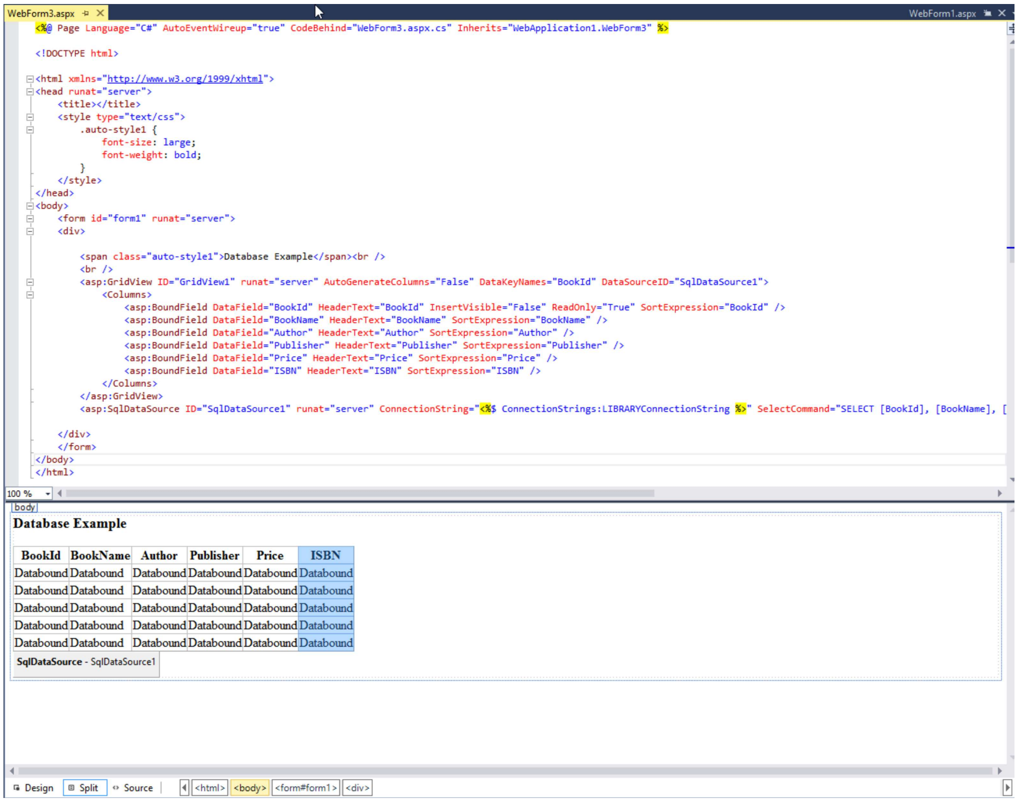
Where will the application get data from?

Connect to any SQL database supported by ADO.NET, such as Microsoft SQL Server, Oracle, or OLEDB.

Specify an ID for the data source:

SqlDataSource1

When finishing the “Wizard”, your .aspx page should look something like this



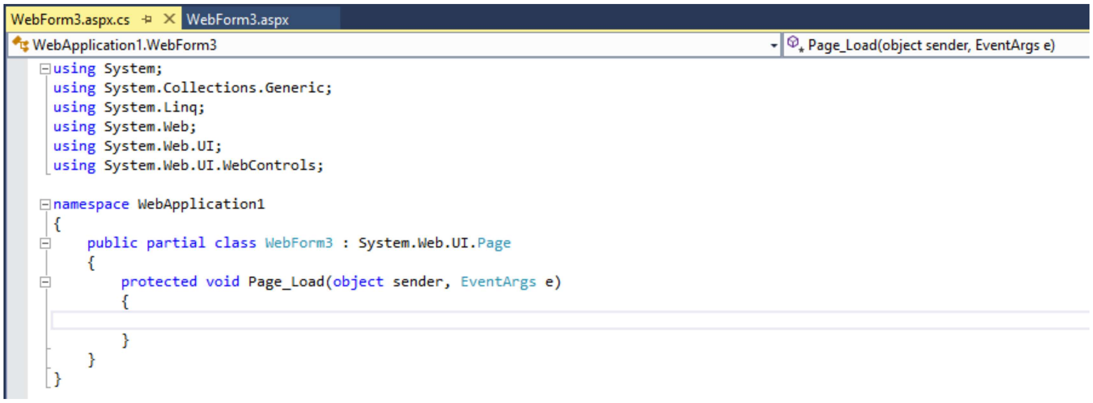
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm3.aspx.cs" Inherits="WebApplication1.WebForm3" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        .auto-style1 {
            font-size: large;
            font-weight: bold;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        <span class="auto-style1">Database Example</span><br />
        <br />
        <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False" DataKeyNames="BookId" DataSourceID="SqlDataSource1">
            <Columns>
                <asp:BoundField DataField="BookId" HeaderText="BookId" InsertVisible="False" ReadOnly="True" SortExpression="BookId" />
                <asp:BoundField DataField="BookName" HeaderText="BookName" SortExpression="BookName" />
                <asp:BoundField DataField="Author" HeaderText="Author" SortExpression="Author" />
                <asp:BoundField DataField="Publisher" HeaderText="Publisher" SortExpression="Publisher" />
                <asp:BoundField DataField="Price" HeaderText="Price" SortExpression="Price" />
                <asp:BoundField DataField="ISBN" HeaderText="ISBN" SortExpression="ISBN" />
            </Columns>
        </asp:GridView>
        <asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%$ ConnectionStrings:LIBRARYConnectionString %>" SelectCommand="SELECT [BookId], [BookName], [

    </div>
    </form>
</body>
</html>
```

Database Example

BookId	BookName	Author	Publisher	Price	ISBN
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound

SqlDataSource - SqlDataSource1

As you see - No Code needed to be written in this example ☺



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication1
{
    public partial class WebForm3 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
    }
}
```

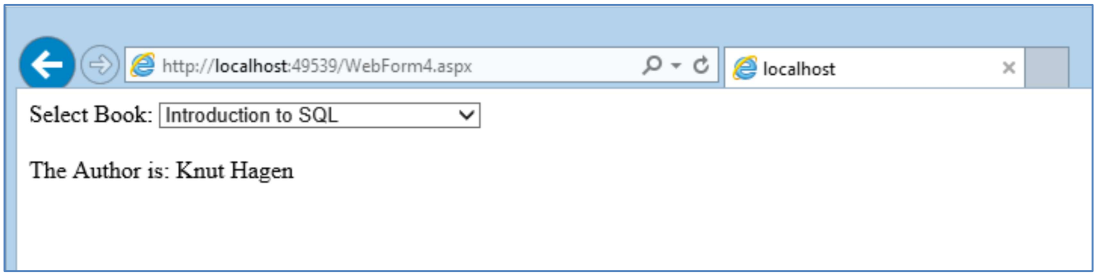
This is OK for quick demos – but for professional applications, you need to use some hardcore ADO.NET or similiar frameworks.

7.4 ADO.NET

ADO.NET (ActiveX Data Object for .NET) is a set of computer software components that programmers can use to access data and data services. It is a part of the base class library that is included with the Microsoft .NET Framework. It is commonly used by programmers to access and modify data stored in relational database systems, though it can also access data in non-relational sources.

7.4.1 Example

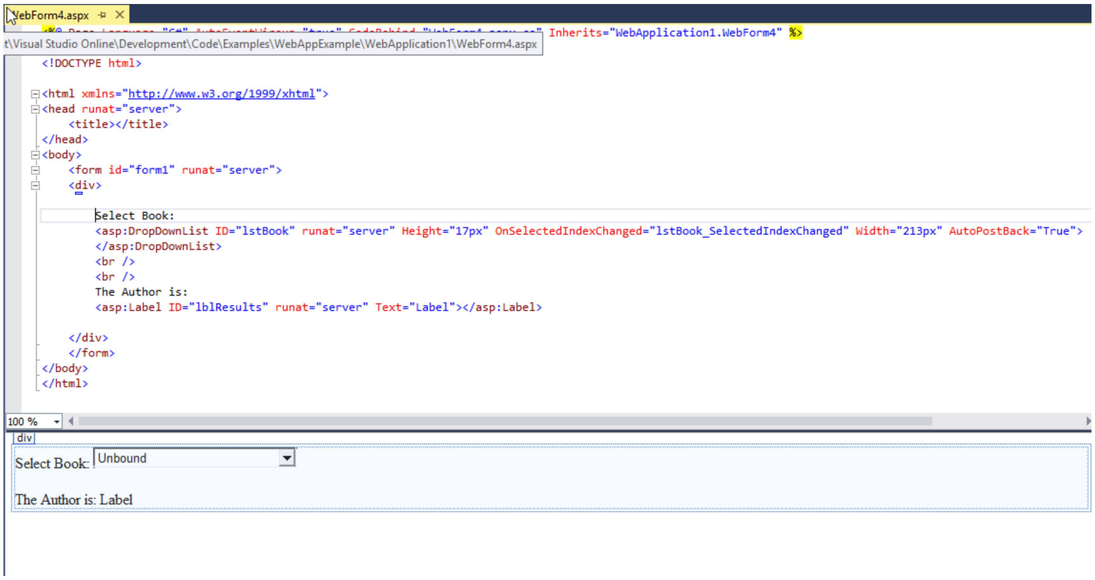
This time: Create everything from scratch using C# code and ADO.NET. Fill a “DropDownList” with Book Names from the Database Print the Author Name based on the selected BookName on the screen, see below.



Select Book: Introduction to SQL

The Author is: Knut Hagen

Your .aspx page should look something like this:

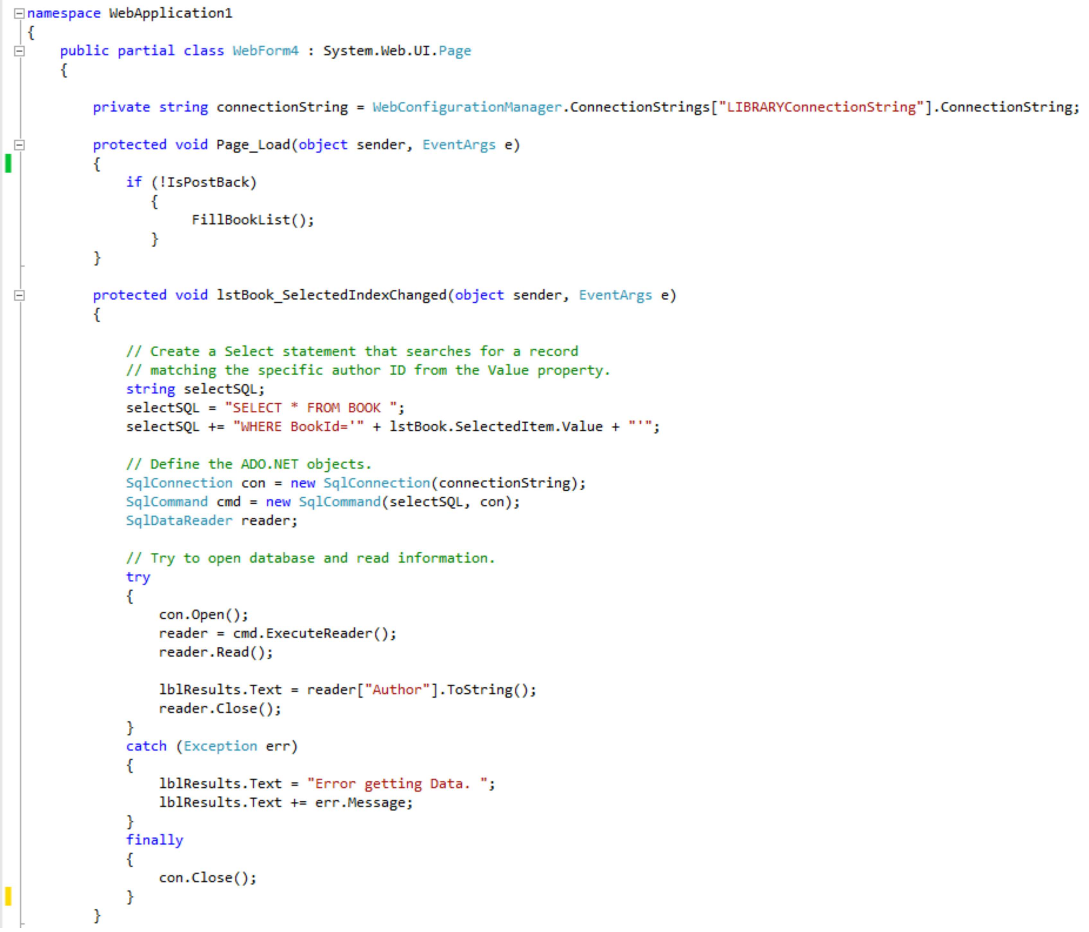

WebForm4.aspx
t\Visual Studio Online\Development\Code\Examples\WebAppExample\WebApplication1\WebForm4.aspx
100 %
div
Unbound

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm4.aspx.cs" Inherits="WebApplication1.WebForm4" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        Select Book:
        <asp:DropDownList ID="lstBook" runat="server" Height="17px" OnSelectedIndexChanged="lstBook_SelectedIndexChanged" Width="213px" AutoPostBack="True">
        </asp:DropDownList>
        <br />
        <br />
        The Author is:
        <asp:Label ID="lblResults" runat="server" Text="Label"></asp:Label>

    </div>
    </form>
</body>
</html>
```

Select Book:

The Author is: Label

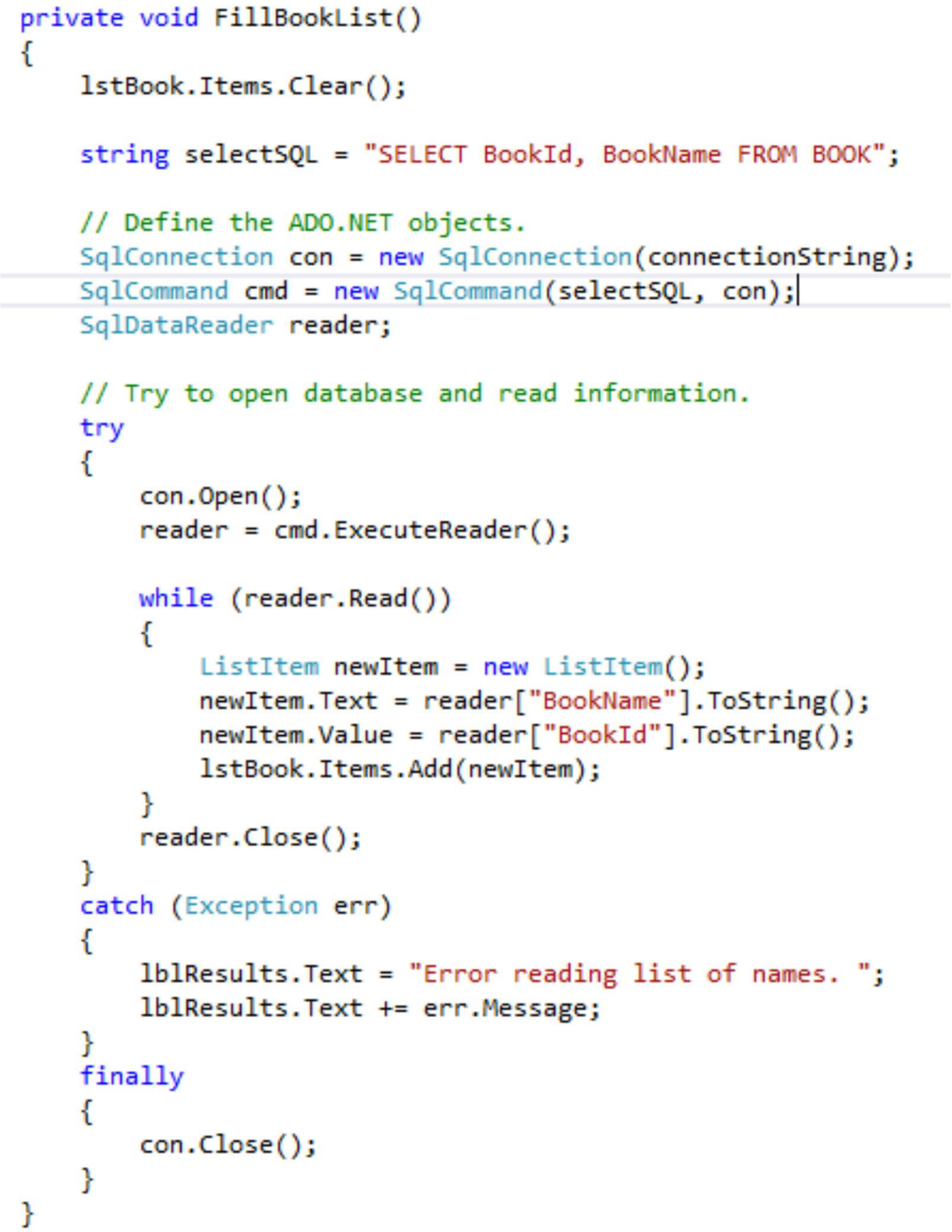
Your .aspx.cs page should look something like this:



```
namespace WebApplication1
{
    public partial class WebForm4 : System.Web.UI.Page
    {

        private string connectionString = WebConfigurationManager.ConnectionStrings["LIBRARYConnectionString"].ConnectionString;

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
               {
                    FillBookList();
               }
        }

        protected void lstBook_SelectedIndexChanged(object sender, EventArgs e)
        {

            // Create a Select statement that searches for a record
            // matching the specific author ID from the Value property.
            string selectSQL;
            selectSQL = "SELECT * FROM BOOK ";
            selectSQL += "WHERE BookId='" + lstBook.SelectedItem.Value + "'";

            // Define the ADO.NET objects.
            SqlConnection con = new SqlConnection(connectionString);
            SqlCommand cmd = new SqlCommand(selectSQL, con);
            SqlDataReader reader;

            // Try to open database and read information.
            try
            {
                con.Open();
                reader = cmd.ExecuteReader();
                reader.Read();

                lblResults.Text = reader["Author"].ToString();
                reader.Close();
            }
            catch (Exception err)
            {
                lblResults.Text = "Error getting Data. ";
                lblResults.Text += err.Message;
            }
            finally
            {
                con.Close();
            }
        }
```

The Page_load() method is executed when the Web Page is loaded. In this simple example we put all the code into the Event Handler for the DropDownList. The code could be improved by creating a separate Class where you put this code into a Method.

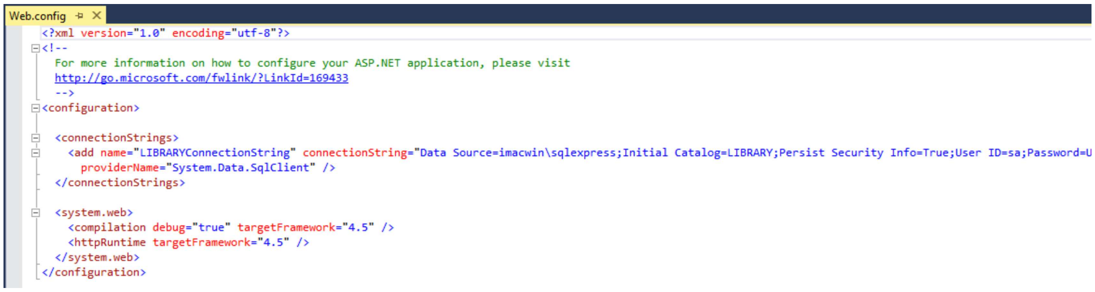
The **FillBookList** method is as follows:



```
private void FillBookList()
{
    lstBook.Items.Clear();

    string selectSQL = "SELECT BookId, BookName FROM BOOK";

    // Define the ADO.NET objects.
    SqlConnection con = new SqlConnection(connectionString);
    SqlCommand cmd = new SqlCommand(selectSQL, con);
    SqlDataReader reader;

    // Try to open database and read information.
    try
    {
        con.Open();
        reader = cmd.ExecuteReader();

        while (reader.Read())
        {
            ListItem newItem = new ListItem();
            newItem.Text = reader["BookName"].ToString();
            newItem.Value = reader["BookId"].ToString();
            lstBook.Items.Add(newItem);
        }
        reader.Close();
    }
    catch (Exception err)
    {
        lblResults.Text = "Error reading list of names. ";
        lblResults.Text += err.Message;
    }
    finally
    {
        con.Close();
    }
}
```

The code could be improved by creating a separate Class where you create this Method.

All kind og configuration should be stored in the Web.config file, including Database Connection Strings, etc.



```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>

  <connectionStrings>
    <add name="LIBRARYConnectionString" connectionString="Data Source=imacwin\sqlexpress;Initial Catalog=LIBRARY;Persist Security Info=True;User ID=sa;Password=U
      providerName="System.Data.SqlClient" />
  </connectionStrings>

  <system.web>
    <compilation debug="true" targetFramework="4.5" />
    <httpRuntime targetFramework="4.5" />
  </system.web>
</configuration>
```