I have created 3 projects, 2 Class Libraries & one Console Application

Git hub link :

https://github.com/Anamika-s/Practice-ADO-Net.git

--------------------------------------------------------

Library which contains Customer Class, CustomerOperations class & app.config file with connectionString

--------------------------------------------------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Cts.Ado.Demo

{

public class Customer

{

public int Id { get; set; }

public string Name { get; set; }

public string Address { get; set; }

public int Qty { get; set; }

public Customer()

{

}

public Customer (int id, string name, string address , int qty)

{

Id = id;

Name = name;

Address = address;

Qty = qty;

}

//int id;

//public int Id

//{

// get { return id; }

// set { id = value; }

//}

//string name;

//public string Name

//{

// get { return name; }

//set { name = value; }

//}

//string address;

//public string Address

//{

// get { return address; }

// set { address = value; }

//}

//int qty;

//public int Qty

//{

// get { return qty; }

// set { qty = value; }

//}

}

}

---------------------------------------------------------------------------

**This file contains all the CRUD operations that you will perform on Customer Class**

**CustosmerOperations.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

// Step 1

using System.Data.SqlClient;

using System.Configuration;

namespace Cts.Ado.Demo

{

public class CustomerOperations

{

// Step2

SqlConnection sqlConnection = null;

// Step 3

SqlCommand sqlCommand = null;

//public CustomerOperations()

//{

// string connectionString = ConfigurationManager.ConnectionStrings["customerConnection"].ToString();

// sqlConnection = new SqlConnection(connectionString);

//}

public SqlConnection GetConnection()

{

string connectionString = ConfigurationManager.ConnectionStrings["customerConnection"].ToString();

sqlConnection = new SqlConnection(connectionString);

return sqlConnection;

}

List<Customer> customerList = null;

public List<Customer> GetCustomers()

{

using (sqlConnection = GetConnection())

{

using (sqlCommand = new SqlCommand("Select \* from customers", sqlConnection))

{

sqlConnection.Open();

SqlDataReader reader = sqlCommand.ExecuteReader();

if (reader.HasRows)

{

customerList = new List<Customer>();

while (reader.Read())

{

Customer customer = new Customer();

customer.Id = (int)reader["id"];

customer.Name = reader["name"].ToString();

customer.Address = reader["address"].ToString();

customer.Qty = (int)reader["qty"];

customerList.Add(customer);

}

}

reader.Close();

sqlConnection.Close();

return customerList;

}

}

if (customerList.Count == 0)

return null;

//sqlCommand.Dispose();

//sqlConnection.Dispose();

}

public bool InsertCustomer(int id, string name, string address, int qty)

{

using(sqlConnection = GetConnection())

{

using (sqlCommand = new SqlCommand("Insert into customers(id, name, address, qty) values (@id,@name,@address,@qty)", sqlConnection))

{

sqlCommand.Parameters.AddWithValue("@id", id);

sqlCommand.Parameters.AddWithValue("@name", name);

sqlCommand.Parameters.AddWithValue("@address", address);

sqlCommand.Parameters.AddWithValue("@qty", qty);

sqlConnection.Open();

int count = sqlCommand.ExecuteNonQuery();

sqlConnection.Close();

if (count > 0) return true;

else return false;

}

}

}

public bool UpdateCustomer(int id, string address, int qty)

{

using (sqlConnection = GetConnection())

{

using (sqlCommand = new SqlCommand("Update customers set address=@address , qty=@qty where id=@id", sqlConnection))

{

sqlCommand.Parameters.AddWithValue("@id", id);

sqlCommand.Parameters.AddWithValue("@address", address);

sqlCommand.Parameters.AddWithValue("@qty", qty);

sqlConnection.Open();

int count = sqlCommand.ExecuteNonQuery();

sqlConnection.Close();

if (count > 0) return true;

else return false;

}

}

}

public bool DeleteCustomer(int id)

{

using (sqlConnection = GetConnection())

{

using (sqlCommand = new SqlCommand("Delete customers where id=@id", sqlConnection))

{

sqlCommand.Parameters.AddWithValue("@id", id);

sqlConnection.Open();

int count = sqlCommand.ExecuteNonQuery();

sqlConnection.Close();

if (count > 0) return true;

else return false;

}

}

}

public Customer FindCustomerById(int id)

{

Customer customer = null;

using (sqlConnection = GetConnection())

{

using (sqlCommand = new SqlCommand("Select \* from customers where id=@id", sqlConnection))

{

sqlCommand.Parameters.AddWithValue("@id", id);

sqlConnection.Open();

SqlDataReader reader = sqlCommand.ExecuteReader();

if (reader.HasRows)

{

customer = new Customer();

customer.Id = (int)reader["id"];

customer.Name = reader["name"].ToString();

customer.Address = reader["address"].ToString();

customer.Qty = (int)reader["qty"];

}

sqlConnection.Close();

}

}

return customer;

}

}

}

App.config file

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

**<connectionStrings>**

**<add name="customerConnection" connectionString="data source=LAPTOP-53S2KQS8;initial catalog=practicedb;integrated security=true"></add>**

**</connectionStrings>**

</configuration>

------------------------------------------------------------------

Console Application

If you are using Library (dll) for keeping ypur classes , then we have to add connectionString in app.config file here also

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />

</startup>

**<connectionStrings>**

**<add name="customerConnection" connectionString="data source=LAPTOP-53S2KQS8;initial catalog=practicedb;integrated security=true"></add>**

**</connectionStrings>**

</configuration>

-----------------------------------------------------

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CustomerConsoleApplication

{

class Program

{

static void Main(string[] args)

{

//// Cts.Ado.Lib.Customer customer = new Cts.Ado.Lib.Customer();

// List<Cts.Ado.Lib.Customer> customerList = new List<Cts.Ado.Lib.Customer>();

// Cts.Ado.Lib.CustomerOperations customerOperations = new Cts.Ado.Lib.CustomerOperations();

// customerList = customerOperations.GetCustomers();

//foreach(Cts.Ado.Lib.Customer customer in customerList)

// {

// Console.WriteLine(customer.Id + "\t" + customer.Name);

// }

// Cts.Ado.Lib.Customer customer = new Cts.Ado.Lib.Customer();

List<Cts.Ado.Demo.Customer> customerList = new List<Cts.Ado.Demo.Customer>();

Cts.Ado.Demo.CustomerOperations customerOperations = new Cts.Ado.Demo.CustomerOperations();

customerList = customerOperations.GetCustomers();

if (customerList != null)

{

foreach (Cts.Ado.Demo.Customer temp in customerList)

{

Console.WriteLine(temp.Id + "\t" + temp.Name);

}

}

else

Console.WriteLine("No Record");

Console.WriteLine("Enter ID");

int id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Quantity");

int qty = int.Parse(Console.ReadLine());

bool isInserted = customerOperations.InsertCustomer(id, name, address, qty);

if (isInserted == true) Console.WriteLine("Record Inserted");

else Console.WriteLine("Record Not Inserted");

// Updattion

Console.WriteLine("Enter ID whode Record to update");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Address");

address = Console.ReadLine();

Console.WriteLine("Enter Quantity");

qty = int.Parse(Console.ReadLine());

bool isUpdated = customerOperations.UpdateCustomer(id, address, qty);

if (isUpdated == true) Console.WriteLine("Record updated");

else Console.WriteLine("Record Not updated");

// Deletion

Console.WriteLine("Enter ID whode Record to delete");

id = int.Parse(Console.ReadLine());

bool isDeleted = customerOperations.DeleteCustomer(id);

if (isDeleted == true) Console.WriteLine("Record deleted");

else Console.WriteLine("Record Not deleted");

// Find Customer By Id

Console.WriteLine("Enter ID whode Record to find");

id = int.Parse(Console.ReadLine());

Cts.Ado.Demo.Customer customer = customerOperations.FindCustomerById(id);

if(customer!=null)

{

Console.WriteLine("Record Found");

Console.WriteLine(customer.Id + "\t" + customer.Name + "\t" + customer.Address+ "\t" + customer.Qty);

}

else

Console.WriteLine("Customer with {0} id does not exist" , id);

}

}

}

---------------------------------------------------------------

**If you are using Collections not database then Customer Class will remain same**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Cts.Ado.Lib

{

public class CustomerOperations

{

List<Customer> customerList = null;

public CustomerOperations()

{

if(customerList == null )

{

customerList = new List<Customer>

{

new Customer() { Id=1, Name ="Deepak", Address="Delhi", Qty=90},

new Customer() { Id=2, Name ="Ajay", Address="NDelhi", Qty=20},

new Customer() { Id=3, Name ="Sagar", Address="Calcutta", Qty=40},

new Customer() { Id=4, Name ="Deepak Garg", Address="ODelhi", Qty=90},

new Customer() { Id=5, Name ="Pradeep", Address="Bombay", Qty=90},

new Customer() { Id=6, Name ="Deepak", Address="Delhi", Qty=90}

};

}

}

public List<Customer> GetCustomers()

{

if (customerList != null)

{

return customerList;

}

else

return null;

}

public bool AddCustomer(Customer customer)

{

if (customer != null)

{

customerList.Add(customer);

return true;

}

else

return false;

}

public bool UpdateCustomer(int id, string address, int qty)

{

bool isUpdated = false;

foreach(Customer temp in customerList)

{

if(temp.Id== id)

{

temp.Address = address;

temp.Qty = qty;

isUpdated = true;

break;

}

}

return isUpdated;

}

public bool DeleteCustomer(int id)

{

bool isDeleted = false;

foreach (Customer temp in customerList)

{

if (temp.Id == id)

{

customerList.Remove(temp);

isDeleted = true;

break;

}

}

return isDeleted;

}

public Customer FindCustomerById(int id)

{

Customer customer = null;

foreach (Customer temp in customerList)

{

if (temp.Id == id)

{

customer = new Customer();

customer = temp;

break;

}

}

return customer;

}

}

}