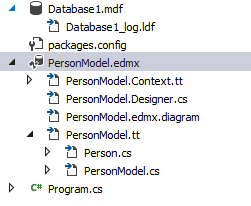
# Intro

## Database first

Add model by database – add new item->data->ADO data model-> generate from database



Model

namespace DataBase\_First

{

using System;

using System.Collections.Generic;

public partial class Person

{

public int Id { get; set; }

public string Name { get; set; }

public Nullable<int> Age { get; set; }

}

}

Context

namespace DataBase\_First

{

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

public partial class PersonEntities : DbContext

{

public PersonEntities()

: base("name=PersonEntities")

{

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

public DbSet<Person> People { get; set; }

}

}

Program.cs

class Program

{

static void Main()

{

var context = new PersonEntities();

foreach (var person in context.People)

{

Console.WriteLine("{0}: {1,-6} - {2} years",person.Id,person.Name,person.Age);

}

}

}

## Adventure work

namespace AW\_DB\_First

{

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

public partial class AdventureWorksLT2012\_DataEntities : DbContext

{

public AdventureWorksLT2012\_DataEntities()

: base("name=AdventureWorksLT2012\_DataEntities")

{

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

public DbSet<Address> Addresses { get; set; }

public DbSet<Customer> Customers { get; set; }

public DbSet<CustomerAddress> CustomerAddresses { get; set; }

public DbSet<SalesOrderDetail> SalesOrderDetails { get; set; }

public DbSet<SalesOrderHeader> SalesOrderHeaders { get; set; }

}

}

static void Main()

{

using (var context = new AdventureWorksLT2012\_DataEntities())

{

var query = from customer in context.Customers

where customer.CustomerAddresses.Any()

orderby customer.FirstName

select new { customer.FirstName, customer.CustomerAddresses };

foreach (var customer in query.Take(10))

{

Console.WriteLine(customer.FirstName);

foreach (var customerAddress in customer.CustomerAddresses)

{

Console.WriteLine("\t" + customerAddress.Address.StateProvince);

}

}

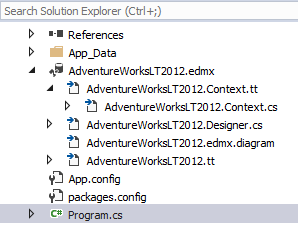
}

}

}

# EDM

## Connection string



class Program

{

static void Main()

{

const string providerName = "System.Data.SqlClient";

const string serverName = @"(LocalDB)\v11.0";

string databasePath = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location)+

@"\App\_Data\AdventureWorksLT2012\_Data.mdf";

// Initialize the connection string builder for the

// underlying provider.

var sqlBuilder = new SqlConnectionStringBuilder

{

DataSource = serverName,

AttachDBFilename = databasePath,

IntegratedSecurity = true

};

// Build the SqlConnection connection string.

string providerString = sqlBuilder.ToString();

// Initialize the EntityConnectionStringBuilder.

var entityBuilder = new EntityConnectionStringBuilder

{

Provider = providerName,

ProviderConnectionString = providerString,

Metadata = @"res://\*/AdventureWorksLT2012.csdl|

res://\*/AdventureWorksLT2012.ssdl|

res://\*/AdventureWorksLT2012.msl"

};

Console.WriteLine(entityBuilder.ToString());

using (var conn = new EntityConnection(entityBuilder.ToString()))

{

conn.Open();

Console.WriteLine("Just testing the connection.");

conn.Close();

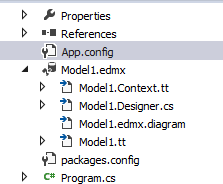
}

Console.ReadKey();

}

}

App config



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

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

<section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=5.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />

</configSections>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />

</startup>

<entityFramework>

<defaultConnectionFactory type="System.Data.Entity.Infrastructure.SqlConnectionFactory, EntityFramework" />

</entityFramework>

<connectionStrings>

<add name="AWLT2012Entities" connectionString="metadata=res://\*/Model1.csdl|res://\*/Model1.ssdl|res://\*/Model1.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=.;initial catalog=ADVENTUREWORKSLT2012\_DATA;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>

</configuration>

class Program

{

static void Main()

{

using (var context = new AWLT2012Entities())

{

foreach (var customer in context.Customers)

{

foreach (var customerAddress in customer.CustomerAddresses)

{

Console.WriteLine(customerAddress.Address.City);

}

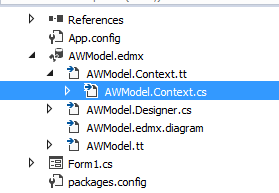
}

}

}

}

## GridView



Form code

public partial class Form1 : Form

{

private readonly AWEntities context;

public Form1()

{

InitializeComponent();

context = new AWEntities();

context.Product.Load();

dataGridView1.DataSource = context.Product.Local.ToBindingList();

}

private void button2\_Click(object sender, EventArgs e)

{

context.SaveChanges();

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

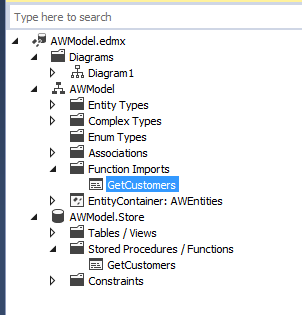
context.Dispose();

}

}

}

## Stored procedure



Form code

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

using (var db = new AWEntities())

{

dataGridView1.DataSource = db.GetCustomers().ToList();

}

}

}

Remove

FormCode

public partial class Form1 : Form

{

readonly AWEntities context;

public Form1()

{

InitializeComponent();

context = new AWEntities();

}

private void Form1\_Load(object sender, EventArgs e)

{

context.Customer.Load();

UpdateDataGridView();

}

private void deleteButton\_Click(object sender, EventArgs e)

{

var customer = dataGridView1.SelectedRows[0].DataBoundItem as Customer;

context.Customer.Remove(customer);

context.SaveChanges();

}

private void UpdateDataGridView()

{

dataGridView1.DataSource = context.Customer.Local.ToBindingList();

}

}

}

## Insert

namespace \_008\_Insert

{

public partial class Form1 : Form

{

DBEntities dbContext;

public Form1()

{

InitializeComponent();

dbContext = new DBEntities();

}

private void Form1\_Load(object sender, EventArgs e)

{

dbContext.PersonSet.Load();

dataGridView1.DataSource = dbContext.PersonSet.Local.ToBindingList();

dataGridView1.Columns[0].Visible = false;

}

private void addButton\_Click(object sender, EventArgs e)

{

var pn = new Person

{

FirstName = txtFName.Text,

LastName = txtLName.Text,

BirthDate = dateTimePicker1.Value

};

dbContext.PersonSet.Add(pn);

dbContext.SaveChanges();

}

private void deleteButton\_Click(object sender, EventArgs e)

{

if (dataGridView1.CurrentRow == null) return;

var current = dataGridView1.CurrentRow.DataBoundItem as Person;

dbContext.PersonSet.Remove(current);

dbContext.SaveChanges();

}

}

}

## One to one

public partial class Form1 : Form

{

OneToOneModelContainer ctx;

public Form1()

{

InitializeComponent();

ctx = new OneToOneModelContainer();

}

private void Form1\_Load(object sender, EventArgs e)

{

LoadData();

#region Set Columns width

dgvEmployee.Columns[0].Width = 30;

dgvEmployee.Columns[1].Width = 78;

dgvEmployee.Columns[2].Width = 75;

dgvEmployeeInf.Columns[0].Width = 30;

dgvEmployeeInf.Columns[1].Width = 78;

dgvEmployeeInf.Columns[2].Width = 80;

dgvEmployeeInf.Columns[3].Width = 100;

#endregion

}

private void btnAdd\_Click(object sender, EventArgs e)

{

var emp = new Employee

{

FirstName = txtFName.Text,

LastName = txtLName.Text,

};

var empInf = new EmployeeInf

{

Age = txtAge.Text,

Gender = txtGender.Text,

Phone = txtPhone.Text

};

emp.EmployeeInf = empInf;

ctx.Employee.Add(emp);

ctx.SaveChanges();

}

private void LoadData()

{

ctx.Employee.Load();

ctx.EmployeeInf.Load();

dgvEmployee.DataSource = ctx.Employee.Local.ToBindingList();

dgvEmployeeInf.DataSource = ctx.EmployeeInf.Local.ToBindingList();

}

private void btnDelete\_Click(object sender, EventArgs e)

{

var emp = dgvEmployee.CurrentRow.DataBoundItem as Employee;

ctx.Employee.Remove(emp);

ctx.SaveChanges();

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

var emp = dgvEmployee.CurrentRow.DataBoundItem as Employee;

emp.FirstName = txtFName.Text;

emp.LastName = txtLName.Text;

EmployeeInf empInf = emp.EmployeeInf;

empInf.Age = txtAge.Text;

empInf.Gender = txtGender.Text;

empInf.Phone = txtPhone.Text;

ctx.SaveChanges();

dgvEmployee.Refresh();

dgvEmployeeInf.Refresh();

}

private void dgvEmployee\_CellClick(object sender, DataGridViewCellEventArgs e)

{

try

{

dgvEmployeeInf.ClearSelection();

dgvEmployeeInf.Rows[dgvEmployee.CurrentRow.Index].Selected = true;

dgvEmployeeInf.CurrentCell = dgvEmployeeInf.SelectedRows[0].Cells[0];

LoadTextBox();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void dgvEmployeeInf\_CellClick(object sender, DataGridViewCellEventArgs e)

{

dgvEmployee.ClearSelection();

dgvEmployee.Rows[dgvEmployeeInf.CurrentRow.Index].Selected = true;

dgvEmployee.CurrentCell = dgvEmployee.SelectedRows[0].Cells[0];

LoadTextBox();

}

private void btnClear\_Click(object sender, EventArgs e)

{

txtFName.Clear();

txtLName.Clear();

txtAge.Clear();

txtGender.Clear();

txtPhone.Clear();

}

//TODO Что Вам не нравится в этом методе?

private void LoadTextBox()

{

txtFName.Text = dgvEmployee.CurrentRow.Cells["FirstName"].Value.ToString();

txtLName.Text = dgvEmployee.CurrentRow.Cells["LastName"].Value.ToString();

txtAge.Text = dgvEmployeeInf.CurrentRow.Cells["Age"].Value.ToString();

txtGender.Text = dgvEmployeeInf.CurrentRow.Cells["Gender"].Value.ToString();

txtPhone.Text = dgvEmployeeInf.CurrentRow.Cells["Phone"].Value.ToString();

}

}

## One to many

public partial class Form1 : Form

{

readonly OneToManyModelContainer ctx;

#region Constructors

public Form1()

{

InitializeComponent();

ctx = new OneToManyModelContainer();

ctx.Models.Local.CollectionChanged += Local\_CollectionChanged;

}

void Local\_CollectionChanged(object sender, System.Collections.Specialized.NotifyCollectionChangedEventArgs e)

{

RefreshModels();

}

private void Form1\_Load(object sender, EventArgs e)

{

ctx.Cars.Load();

ctx.Models.Load();

dgvCars.DataSource = ctx.Cars.Local.ToBindingList();

FillCarTextBoxes();

RefreshModels();

}

#endregion

#region EventHandlers

#region Cars

private void carAddButton\_Click(object sender, EventArgs e)

{

var car = new Car

{

Factory = txtFactory.Text,

Country = txtCountry.Text

};

ctx.Cars.Add(car);

ctx.SaveChanges();

}

private void carUpdateButton\_Click(object sender, EventArgs e)

{

if (dgvCars.CurrentRow != null)

{

var car = dgvCars.CurrentRow.DataBoundItem as Car;

if (car != null)

{

car.Factory = txtFactory.Text;

car.Country = txtCountry.Text;

}

}

ctx.SaveChanges();

dgvCars.Refresh();

}

private void carDeleteButton\_Click(object sender, EventArgs e)

{

if (dgvCars.CurrentRow != null)

{

var car = dgvCars.CurrentRow.DataBoundItem as Car;

ctx.Cars.Remove(car);

}

ctx.SaveChanges();

ClearTextBoxes(carsGroupBox);

}

private void dgvCars\_CellClick(object sender, DataGridViewCellEventArgs e)

{

FillCarTextBoxes();

RefreshModels();

}

#endregion

#region Models

private void modelAddButton\_Click(object sender, EventArgs e)

{

if (dgvCars.CurrentRow == null) return;

var car = dgvCars.CurrentRow.DataBoundItem as Car;

var model = new Model

{

Name = txtName.Text,

Color = txtColor.Text,

Engine = txtEngine.Text

};

car.Models.Add(model);

ctx.SaveChanges();

}

private void modelUpdateButton\_Click(object sender, EventArgs e)

{

if (dgvModel.CurrentRow == null) return;

var model = dgvModel.CurrentRow.DataBoundItem as Model;

model.Name = txtName.Text;

model.Color = txtColor.Text;

model.Engine = txtEngine.Text;

ctx.SaveChanges();

dgvModel.Refresh();

}

private void modelDeleteButton\_Click(object sender, EventArgs e)

{

if (dgvModel.CurrentRow != null)

{

var model = dgvModel.CurrentRow.DataBoundItem as Model;

if (model != null)

{

ctx.Models.Remove(model);

ctx.SaveChanges();

}

}

}

private void dgvModel\_CellClick(object sender, DataGridViewCellEventArgs e)

{

FillModelTextBoxes();

}

#endregion

private void btnClear\_Click(object sender, EventArgs e)

{

ClearTextBoxes(modelsGroupBox);

ClearTextBoxes(carsGroupBox);

}

#endregion

#region Helpers

private void ClearTextBoxes(Control gBox)

{

foreach (var textBox in gBox.Controls.OfType<TextBox>())

textBox.Clear();

}

private void RefreshModels()

{

if (dgvCars.CurrentRow != null)

{

var currentCar = dgvCars.CurrentRow.DataBoundItem as Car;

if (currentCar != null) dgvModel.DataSource = currentCar.Models.ToList();

}

FillModelTextBoxes();

}

## Many to many

public partial class Form1 : Form

{

readonly ManyToManyModelContainer context;

public Form1()

{

InitializeComponent();

context = new ManyToManyModelContainer();

}

private void Form1\_Load(object sender, EventArgs e)

{

LoadStudents();

LoadCourses();

#region Set Columns width

dgvStudents.Columns[0].Width = 25;

dgvStudents.Columns[1].Width = 88;

dgvStudents.Columns[2].Width = 88;

dgvCourses.Columns[0].Width = 25;

dgvCourses.Columns[1].Width = 160;

#endregion

}

private void LoadStudents()

{

dgvStudents.DataSource = context.Students.ToList();

}

private void LoadCourses()

{

dgvCourses.DataSource = context.Courses.ToList();

}

//Добавление 3х новых студентов и 3 новых курса.

private void btnRun\_Click(object sender, EventArgs e)

{

var student1 = new Student { FirstName = "Александр", LastName = "Гофман" };

var student2 = new Student { FirstName = "Алексей", LastName = "Иванов" };

var student3 = new Student { FirstName = "Василий", LastName = "Петрук" };

var course1 = new Course { Name = "WCF" };

var course2 = new Course { Name = "WPF" };

var course3 = new Course { Name = "JavaScript" };

student1.Courses.Add(course1);

student1.Courses.Add(course2);

student2.Courses.Add(course3);

student2.Courses.Add(course2);

student3.Courses.Add(course1);

student3.Courses.Add(course3);

context.Students.Add(student1);

context.Students.Add(student2);

context.Students.Add(student3);

context.SaveChanges();

LoadCourses();

LoadStudents();

}

private void dgvStudents\_CellClick(object sender, DataGridViewCellEventArgs e)

{

dgvCourses.ClearSelection();

if (dgvStudents.CurrentRow != null)

{

var studentCources = ((Student)dgvStudents.CurrentRow.DataBoundItem)

.Courses;

var studentCourcesRows = from DataGridViewRow row in dgvCourses.Rows

let cource = row.DataBoundItem as Course

where studentCources.Contains(cource)

select row;

foreach (DataGridViewRow row in studentCourcesRows)

row.Selected = true;

}

}

//TODO Дома: Переписать без этого ужаса! (по аналогии с методом dgvStudents\_CellClick

private void dgvCourses\_CellClick(object sender, DataGridViewCellEventArgs e)

{

dgvStudents.ClearSelection();

int courseId = (int)dgvCourses.CurrentRow.Cells["Id"].Value;

var query = context.Students.Where(s => s.Courses.Any(c => c.Id == courseId));

for (int i = 0; i < dgvStudents.Rows.Count; i++)

{

foreach (var item in query)

{

if (dgvStudents.Rows[i].Cells[0].Value.ToString() == item.Id.ToString())

{

dgvStudents.Rows[i].Selected = true;

}

}

}

}

}