**Lab – Repository Pattern sample**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Linq.Expressions;

namespace DataAccessLayer

{

public interface IRepository<T> : IDisposable

{

void Insert(T entity);

T GetSingle(Func<T, bool> where, params Expression<Func<T, object>>[] navigationProperties);

}

}

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Linq.Expressions;

namespace DataAccessLayer

{

public class Repository<T> : IRepository<T> where T : class

{

protected DbContext context;

protected DbSet<T> dbset;

public Repository(DbContext context)

{

this.context = context;

dbset = context.Set<T>();

}

public void Insert(T entity)

{

context.Entry(entity).State = EntityState.Added;

context.SaveChanges();

}

public T GetSingle(Func<T, bool> where, params Expression<Func<T, object>>[] navigationProperties)

{

T item = null;

IQueryable<T> dbQuery = context.Set<T>();

//Apply eager loading

foreach (Expression<Func<T, object>> navigationProperty in navigationProperties)

dbQuery = dbQuery.Include<T, object>(navigationProperty);

item = dbQuery

.AsNoTracking() //Don't track any changes for the selected item

.FirstOrDefault(where); //Apply where clause

return item;

}

namespace DataAccessLayer

{

public interface IStandardRepository : IRepository<Standard> { }

public class StandardRepository : Repository<Standard>, IStandardRepository

{

public StandardRepository() : base(new SchoolDBEntities()) { }

}

}

using DataAccessLayer;

using System.Collections.Generic;

namespace BusinessLayer

{

public interface IBusinessLayer

{

IList<Standard> GetAllStandards();

Standard GetStandardByID(int id);

Standard GetStandardByName(string name);

void AddStandard(Standard standard);

IList<Student> GetAllStudents();

Student GetStudentByID(int id);

Student GetStudentByName(string name);

void AddStudent(Student student);

void UpdateStudent(Student student);

void RemoveStudent(Student student);

}

}

using DataAccessLayer;

using System.Collections.Generic;

using System.Linq;

using System;

namespace BusinessLayer

{

public class BusinessLayer : IBusinessLayer

{

private readonly IStandardRepository \_standardRepository;

public BusinessLayer()

{

\_standardRepository = new StandardRepository();

}

public void AddStandard(Standard standard)

{

\_standardRepository.Insert(standard);

}

public Standard GetStandardByIDWithStudents(int id)

{

return \_standardRepository.GetSingle(

s => s.StandardId == id,

s => s.Students);

}

public Standard GetStandardByName(string name)

{

return \_standardRepository.GetSingle(

s => s.StandardName.Equals(name),

s => s.Students);

}

}

}

using DataAccessLayer;

using System;

using System.Collections.Generic;

namespace Client

{

class Program

{

//----------------------------------------- Slots ------

static BusinessLayer.BusinessLayer bl = new BusinessLayer.BusinessLayer();

SearchStandardStudents();

break;

------------------- SearchStandardStudents ------

private static void SearchStandardStudents()

{

Console.Write("Enter Standard ID: ");

ReadChoice();

Standard s = bl.GetStandardByIDWithStudents(choice);

if (s == null)

{

Console.WriteLine("Standard not found!");

return;

}

else if (s.Students == null || s.Students.Count == 0)

{

Console.WriteLine("This Standard has no Students!");

return;

}

Console.WriteLine("\nContaining Students: {0}", s.Students.Count);

foreach (Student student in s.Students)

Console.WriteLine("- " + student.StudentName);