# Lab ToDoApp – Step5 - Datalayer

Übungsdauer : 60 Minuten

## Overview

In dieser Übung wird nun das Entitäten Model für eine Datenbank vorbereitet und mit dem Code-First Ansatz über das Entity Framework in eine Datenbank geschrieben. Dabei wird der n:m Ansatz für Kategorie und ToDo Datenbank-technisch korrekt abgebildet

## Ziel

* EF – Code First
* Entity Framework Context

## Schritte

Zuerst muss das Projekt für das EntityFramework vorbereitet werden und ein Datenbank-Context als Repository erstellt werden.

1. **Öffnen sie die Solution**
2. **Entityframework über NuGet installieren**
   * Klicken sie auf das Projekt „ToDoApp.Business“ und wählen sie den Reference Ordner mit der rechten Maustaste an und wählen das Menü „Manage NuGet Packages“
   * Klicken sie im oberen Bereich auf „Browse“
   * Wählen sie das NuGet Package „EntityFramework“ aus
   * Klicken sie auf Install und bestätigen sie alle Meldungen.
   * Wiederholen sie die Schritte für das Projekt „ToDoApp.Client
3. **Entity-Framework Context erstellen**
   * Klicken sie im Projekt „ToDoApp.Business“ auf die Projektdatei mit der rechten Maustaste
   * Wählen sie den Menüpunkt „Add/New Item“ aus
   * Wählen sie die Vorlage „Class“ aus und geben sie der Klasse den Namen „ToDoRepositoryEDM“
   * Kennzeichnen sie die Klasse mit „public“
   * Leiten sie die Klasse von DbContext ab
   * Fügen sie folgende Zeile für die Abfrage aller ToDo´s ein:

public DbSet<ToDo> ToDoes { get; set; }

* + Fügen sie folgende Zeile für die Abfrage aller ToDoCategories ein:

public DbSet<ToDoCategory> ToDoCategories { get; set; }

* + Das neue Repository soll die gleichen CRUD Methoden wie das erste Repository bekommen. Dazu fügen sie zuerst die GetAll Methode ein:

public IQueryable<ToDo> GetAll()

{

return ToDoes;

}

* + In der Create Methode wird ein neues Objekt ToDo erstellt und dieses an den Context übermittelt. Anschließend soll der Context in die Datenbank gespeichert werden:

public ToDo Create()

{

ToDo myResult = new ToDo("Neue Aufgabe");

ToDoes.Add(myResult);

this.SaveChanges();

return myResult;

}

* + Die Update und Delete Methoden müssen analog umgebaut werden.

Tipp: Die Update Methode muss nur den Context speichern also SaveChanges()!

* + Nun müssen die Entitäten noch mit einem Attribut versehen werden, um den EF mitzuteilen wie und wer der Primärschlüssel ist.
  + Öffnen sie die Datei ToDo.cs
  + Fügen sie folgendes Attribut über die ID Spalte ein:

[Key]

[DatabaseGeneratedAttribute(DatabaseGeneratedOption.Identity)]

## Schritte

Im zweiten Schritt wird das neue EDM-Repository mit dem Client verdrahtet. Dazu wird eine Such-Command und eine SuchBox in die WPF Maske eingebunden. Ebenso werden alle schon vorhandenen Commandos auf das neue Repositorie umgebaut

1. **Öffnen sie die Solution**
2. **MainViewModel für neues Repository umbauen**
   * Klicken sie auf das Projekt „ToDoApp.Client“
   * Öffnen sie die Datei MainViewModel.cs
   * Suchen sie nach allen Stellen in dem „ToDoRepository“ angegeben wird und ersetzten dieses mit „ToDoRepositoryEDM“
   * Ergänzen sie alle Stellen an den GetAll() verwendet wurde mit einer .ToList() Anweisung.
3. **Suchfunktion implementieren**
   * Öffnen sie die Datei MainViewModel.cs
   * Fügen sie die Eigenschaft „Suchwort“ als Eigenschaft wie folgt ein:

private string \_Suchwort;

public string Suchwort

{

get { return \_Suchwort; }

set { \_Suchwort = value; RaisePropertyChange("Suchwort"); }

}

* + Fügen sie eine Eigenschaft für das Such-Kommando ein:

public ICommand SearchCommand { get; set; }

* + Erstellen sie im Konstruktor ein Such-Kommando Objekt:

SearchCommand = new StandardCommand(Search);

* + Fügen sie eine Suchfunktion ein:

private void Search(object parameter)

{

}

* + Schreiben sie eine LINQ – Query für das suchen von ToDo-Aufgaben:

var myQuery = from myToDo in \_ToDoRepository.GetAll()

where myToDo.Title.Contains(this.Suchwort)

select myToDo;

this.DataList = myQuery.ToList();

Achten sie auch noch auf das Exception Handling wie in den anderen Methoden.

* + Kompilieren sie ihre Anwendung
  + Öffnen sie die Datei „MainView.xaml“ um das neue Kommando und die Suchbox zu erstellen
  + Fügen sie auf den View eine TextBox oben rechts ein und daneben einen Button
  + Selektieren sie die Textbox
  + Klicken sie im Eigenschaftenfenster neben die Eigenschaft Text auf das schwarze Viereck und dann auf den Menüpunkt „Databinding erstellen“
  + Wählen sie die Eigenschaft „Suchwort“ aus
  + Selektieren sie nun den Button
  + Wählen sie die Eigenschaft „Command“ aus mit dem Context Menü um ein Databinding zu erstellen und wählen sie die Eigenschaft „SearchCommand“ aus.
  + Testen sie ihre Anwendung