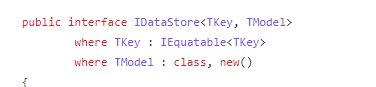
**Generic data store**

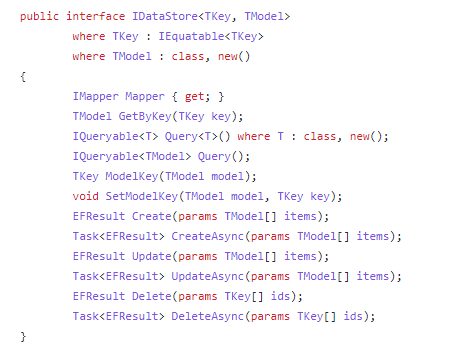
<https://www.youtube.com/watch?v=0FnJeikULJU&ab_channel=NDCConferences>

<https://github.com/donwibier/GenericAbstractFun/blob/master/DXWeb.RefactorDemo/Models/EFStore.cs>



TKey is used as an generic type for the primary key, it might be int/guild/string.Beeing a primary key, it must be unique, threrefore it must be a type that has Equals implemented, meaning it must implement IEqutable interface

TModel is the DTO that should exist the data store, it must be a class and have an empty constructor



Any data store should be able to perform the following operations

*GetByKey - gets the primary key and returins TModel, which is the DTO*

ex:

EmployeeDTO employee = employeeStore.GetById(userId);

*Query<T> - gets the IQueryable<T>, in order to perform specific operations on it before executing the query;*

ex:

var employees = employeeStore.Query<Employee>().Where(etc etc);

//Type of T should be a database entity class

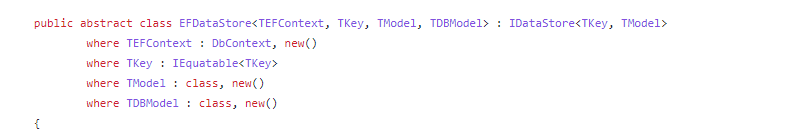
*Query() - gets the IQueryable<TModel>, directly the DTO*

*ModelKey(TModel) returns the primary key for that DTO*

*SetModelKey(TModel,TKey) - sets the primary key*

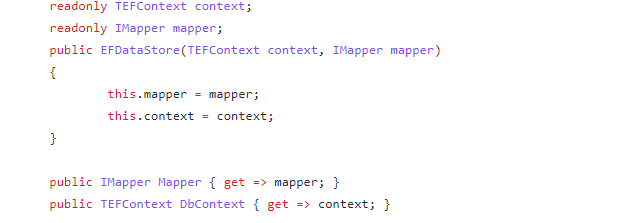
*Create,Update,Delete - bulk operations*

**Implementation**

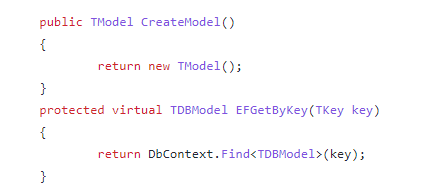


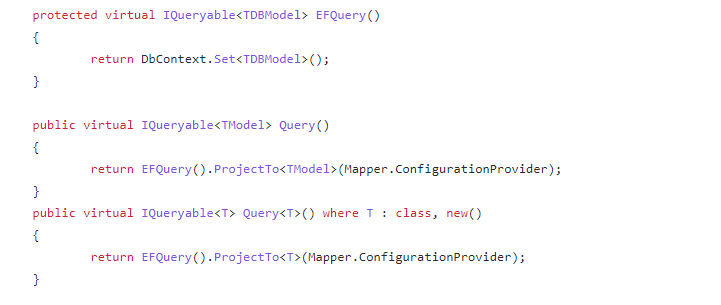
Specific data store implementation for an application using EntityFramework core as ORM

TDBModel is the entity class used for DB tables



Dependency injection for the mapper and context, every EF Core data store implementation should have these 2





ProjectTo<TModel>(Mapper.configurationProvider) comes from *using AutoMapper.QueryableExtensions;*

Using these, you can easly query using automapper

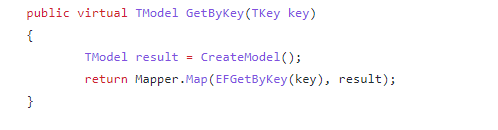
Example

In your mapper profile you have the following:

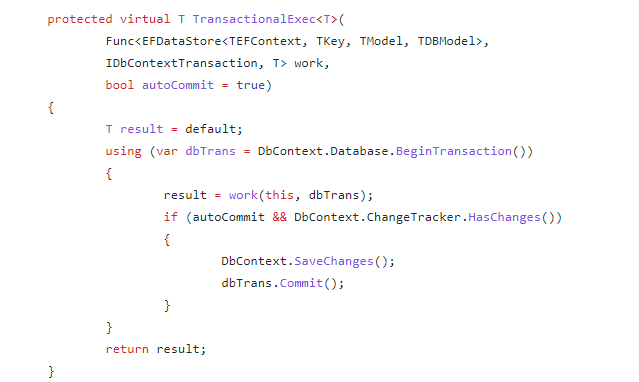
CreateMap<Employee,EmployeeDTO)

.ForMember(dt => dt.DepartmentName,opt => opt.MapFrom(src => src.department.Name))

Using the ProjectTo, ef core does the needed include and returns it as a DTO



**Generic transactions**

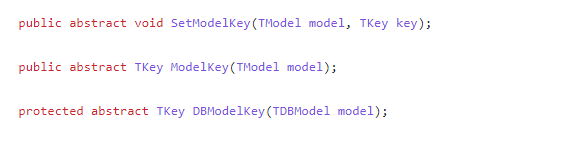
****

Has two parameters : a func that has 2 params and a boolean autoCommit

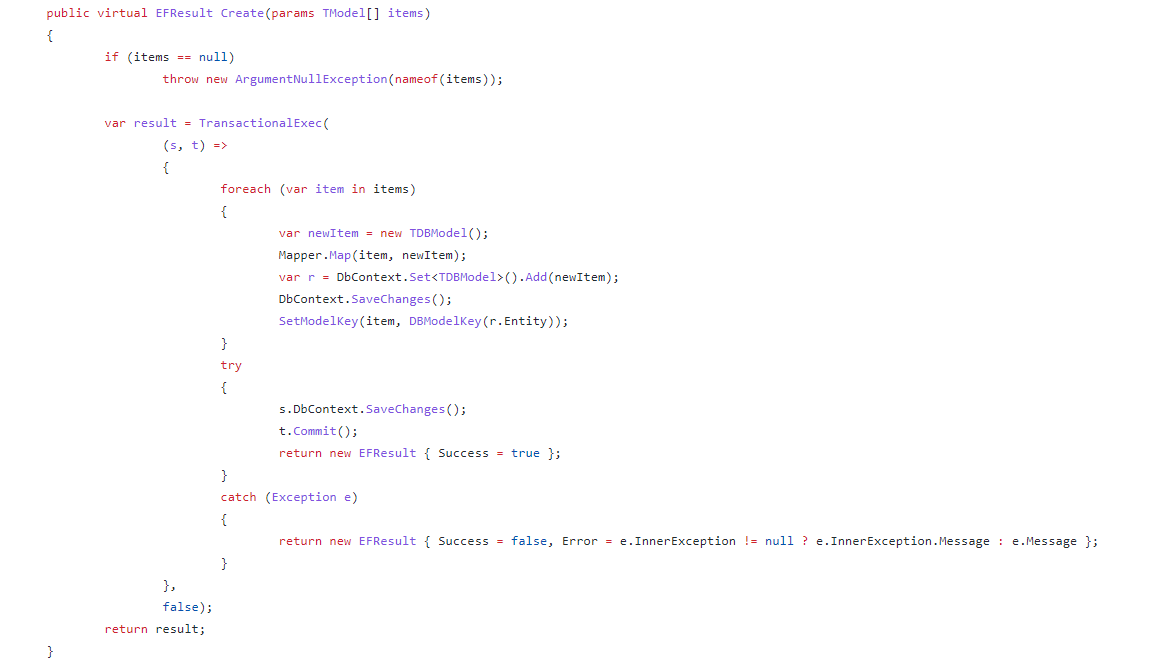
The func has two paramters and returns T

Implementaiton is just openenning a transaction, runing the function from paratmer - work - and storing it's result. IfAutoCommit is true and changes have been made, we saveChanges and roll thoses comits;

This method can be used whenever you have to run code that should happen in a single transaction;



These are left abstract, because at the moment we don't know the primary key, for example EmployeeDataStore might have EmployeeId as it's PK, CustomerDataStore might have CustomerId etce tc

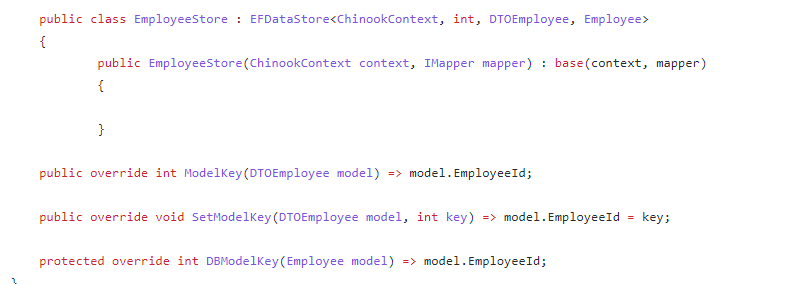


Creates a list of Items, and it must be in a single transaction

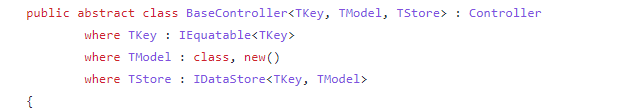
(s,t) are the parameters , a function that has a parrameter s of type EfDataStore and the second t is of type IDBContextTransaction.

it just loops thourh the items and adds them to the context, i don't understand why he calls DbContext.SaveChanges and t.DbContext.SaveChanges and t.Commit because these two are already called inside the implementation of TransactionalExec

**Example of store implementation**

****

**Base controller**

****

For every entity, Employee,Customer,Invoice etc you need a controller handling those entities

TStore is the store you need, for example EmployeesController has EmployeeDataStore as it's data access