**Mapovani entit**

Jak vytvorit mapovaci tabulku :

Tabulka musi mit atributy

[EnumTable] => pokud jde o ciselnik

[DaoFactory(DaoType = typeof(MarketDao))] => pokud chceme k tabulce priradit dao tridu

[TableName(Owner = MediaDataDataModel.MediaSchema)] => vzdy

Tabulku musime pridat do DataModelu:

public Table<Market> Market => GetTable<Market>();

Jak vytvorit dao tridu:

Poznamka 15 February 2018 Na vytvoreni Dao tridy potrebujeme mit hotovou Table. Priklad: ve zmenovadle byla tabulka Market hotova ale nebylo dao. Vytvoril jsem pro tuto tabulku dao tridu MarketDao

public class MarketDao : EntityDaoBase<Market, MediaDataDataModel, byte, Market>.

Ta ma dva konstruktory:

public MarketDao(string dbAlias, IsolationLevel isolationLevel)

: base(dbAlias, isolationLevel)

{

}

public MarketDao(string dbAlias)

: base(dbAlias)

{

}

Aby byla dao trida funkcni musi se do tabulky patrici k dao tride pridat metadata atribut:

[DaoFactory(DaoType = typeof(MarketDao))]

* K mapovani entit jsem se dostal pri lokalizaci ciselniku. Mapovani entity musime provest vzdy kdyz chceme nad databazovou tabulkou ve ktere je entita volat jakykoli dotaz .
* Tabulku PartName je potreba **namapovat** abychom mohli v daotridach volat prez tabulky dotazy. Mapovane entity jsou samozrejme ve slozce Entities, stejne jako jejich dao tridy. Jejich dataContracty jsou ve slozce Entites/MIR.Common2.PrintStorage.DataContracts.
* Kazda namapovana tabulka se musi pridat do dataModelu . V tomto pripade je to trida**: PrintStorageDataModel**

public Table<PartName> PartName => GetTable<PartName>();

* Dao tridy jsou ve sl. Entities pod PrintStorage\Dao\Dbo
* A mapovaci tridyentit : pod PrintStorage\Dbo
* Nad mapovanou tabulku je potreba pridat **BLToolkit znacky** . :

namespace MIR.Entities.PrintStorage.Dbo

{

**=> [TableName]**

**=> [LocalizedEnumTable(typeof(LocalizedPartName))] => definuje , ze se jedna o tabulku s lokalizaci (jen ciselniky)**

public class PartName : **LightDatabaseEntityByteKey<PartName>, IEnumerableTableWithDescriptionAndOrder<byte> => rozhrani jen u ciselniku**

**trida dedi podle toho jestli ma identitu a inkrementaci , byte je v tomto pripade proto, ze primarni klic tabulky je typu tinyint coz je byte (8 bitu)**

{

public string Name { get; set; }

public bool IsSheet { get; set; }

public bool IsCover { get; set; }

public bool PictureMatching { get; set; }

**tady neni [mapIgnore] protoze tento sloupec jsme pridali pomoci scriptu kvuli ciselniku**

public string Description { get; set; }

[MapIgnore]

public byte IdValue => Id;

[MapIgnore] **=> definuje , ze sloupec neni v tabulce PartName prezto je namapovany**

public string NameValue => Name;

[MapIgnore]

public byte OrderValue { get; set; }

[MapIgnore]

public string DescriptionValue

{

get => Description;

set => Description = value;

}

public class LocalizedPartName : DefaultNamingConventionEnumTableMappingByteKey<PartName>

**definuje , ze se ma hledat preklad v tabulce ktera se jmenuje jako tato a na konci ma Translation napr PartName a PartNameTranslation.**

**(jen u ciselniku)**

{

}

}

* Po namapovani a pridani do tridy modelu by mela byt mapovana trida viditelna na modelu v dao tride .
* Preklad dostaneme tak ze si normalni cestou na daoTride vytahneme datacontract ktery potrebujeme a potom v servisni akci zmenime sloupec ktery se ma prelozit pomoci volani metody **GetValueId( nejakeId ). DescriptionValue** nez vratime response ze servisni akce

Predpoklada se uz vytvoreny sloupec description v databazi. :D

Pro ciselniky : Sloupec v tabulkach JmenoTranslation se musi jmenovat LocalizationLanguageId.

protected override ListResponse<PublicationDataContract> ExecuteInternal(GetPublicationsRequest request)

{

var items = m\_publicationDao.GetPublications(request.AdvertisedFrom, request.AdvertisedTo, request.MediumId, request.PublicationStatusId , request.ProcessorId,request.Comment);

foreach (var item in items)

{

item.Status = m\_publicationStatuts.**GetValueById(item.PublicationStatusId).DescriptionValue;**

**item.Status je ten prekladany sloupec.Timto dostameme prelozeny sloupec . V metode vlastne predame id radku na ktere lezi slovo ktere chceme prekladat , a vrati se jeho descriptionValue. To je v pripade ,ze je vice jazyku hodnota sloupce description v tabulce necoTranslation.**

}

return new ListResponse<PublicationDataContract>(items);

}

* **Mitroz** mi opravil mapovani takto: Puvodne jsem pomoci dotazu v MediumDao chtel MediumDataContract ktery mel data ze tri tabulek a jeste je tridil pomoci logiky v metode v MediumDao. Ta byla prilis dlouha a neprehledna . Mitroz namapoval entitu Medium ktere pridal navic Vlastnost MediumPublicationPeriodicity a MediumScanningPresset coz jsou ve skutecnosti dve tabulky v databazi . Takto mame v jedine namapovane tride to , co by jinak bylo ve trech .

[TableName]

[DaoFactory(DaoType = typeof(MediumDao))]

public class Medium : LightDatabaseEntityShortKey<Medium>, IAuditable

{

[Nullable]

public string OriginalId { get; set; }

public string Name { get; set; }

[Nullable]

public string Comment { get; set; }

[Nullable]

public byte? ProcessorId { get; set; }

[Nullable]

public short? PageWidth { get; set; }

[Nullable]

public short? PageHeight { get; set; }

public DateTime Created { get; set; }

public byte CreatedBy { get; set; }

public DateTime? Modified { get; set; }

public byte? ModifiedBy { get; set; }

**Pouziti mapperu:**

[Association(CanBeNull = true, ThisKey = nameof(Id), OtherKey = "MediumId")]

public MediumPublicationPeriodicity Periodicity { get; set; }

[Association(CanBeNull = true, ThisKey = nameof(Id), OtherKey = "MediumId")]

public MediumScanningPreset ScanningPreset { get; set; }

}

**Mappery:**

Mapper je jednoducha trida ktera pomoci metody map() , nebo createMap() prevede entitu na dataContract.

**Priklad ze skoleni pouzito pri UnitTestech**:

* Udelame slozku v Core
* Vytvorili jsme public class GalaxyDataContractMapper : IMapper<Galaxie, GalaxyDataContract>
* Rodicovska trida prebira dva parametry , jeden vstup a jeden vystup
* Maper je jednoducha zalezitost ktera prevadi pomoci metodymap()

public GalaxyDataContract Map(Galaxie obj)

{

return new GalaxyDataContract(obj.Id, obj.Jmeno, obj.PolohaX, obj.PolohaY, obj.PolohaZ);

}

* Vraci vystupni objekt.

**Pouzite v Scanovadle**: Mapper nam pomuze namapovat vice tabulek a vlozit je do jedne mapovane tabulky . Potrebujeme pomoci mapperu vytvorit datacontract z entit v tabulkach PublicationPeriodicity a MediumScanningPreset a vlozit je do mapovane tridy Medium. Takze vlastne Medium bude mit na sobe vlastnosti Peridicity a ScanningPreset ktere v db vubec nema, navic na techto vlastnostech jsou dalsi podvlastnosti podle toho jake sloupce maji . pridane tabulky.

* Vytvorime slozku Mappers v Core a v ni bude trida Scanning2CoreMappersConfiguration : MappingConfiguratorBase:

public class Scanning2CoreMappersConfiguration : MappingConfiguratorBase

{

protected override void ConfigureInternal()

{

CreateMap<Medium, MediumDataContract>();

CreateMap<MediumPublicationPeriodicity, PublicationPeriodicityDataContract>()

.ForMember(d => d.PeriodicityTypeId, x => x.MapFrom(v => v.PeriodicityType))

.ForMember(d => d.DayOfMonth, x => x.MapFrom(v => v.BasicFrequency));

CreateMap<MediumScanningPreset, MediumScanningPresetDataContract>();

}

}

* Aby bylo mozne mapper pouzit musi se resolvnout pomoci CastleContaineru

var mappingConfiguratior = m\_globalContainer.Resolve<DependencyMappingConfigurator>();

mappingConfiguratior.Configure(); // vola metodu v MediaResearch Frameworku

* V Core je slozka Installers a v ni je MappingInstaller : WindsorInstaller. Zaregistrujeme ConfigurationBase a kazdy IMapper

namespace MIR.Media.Changing2.Core.Installers

{

public class MappingInstaller : IWindsorInstaller

{

public void Install(IWindsorContainer container, IConfigurationStore store)

{

container.Register(Component.For<MappingConfiguratorBase>().ImplementedBy<Changing2CoreMappersConfiguration>().LifestyleSingleton());

container.Register(Component.For<IMapper<GetDataRequest, MotivletFilterParams>>().ImplementedBy<SimpleMapper<GetDataRequest,MotivletFilterParams>>());

}

}

}

* V mapovane tride muzeme ted pridat Vlastnosti ziskane pomoci mapperu. // BLToolkit dokaze sam vygenerovat join na zaklade atributu Association

Propojeno na zaklade Id == MediumId.

[Association(CanBeNull = true, ThisKey = nameof(Id), OtherKey = "MediumId")]

public MediumPublicationPeriodicity Periodicity { get; set; }

[Association(CanBeNull = true, ThisKey = nameof(Id), OtherKey = "MediumId")]

public MediumScanningPreset ScanningPreset { get; set; }

* Ve tride MediumDao ted staci ziskat entitu Medium ktera uz ma nove Vlastnosti

using (var model = CreateDbContext())

{

var query = from m in model.Medium

orderby m.Name

select new

{

Medium = m,

Periocitidy = m.Periodicity,

Preset = m.ScanningPreset

};

var data = query.ToList();

List<Medium> result = new List<Medium>();

foreach (var dato in data)

{

dato.Medium.Periodicity = dato.Periocitidy;

dato.Medium.ScanningPreset = dato.Preset;

result.Add(dato.Medium);

}

return result;

* Mapper se musi zaregistrovat v MappingInstalleru:

namespace MIR.PrintStorage.Scanning2.Core.Installers

{

public class MappingInstaller : IWindsorInstaller

{

public void Install(IWindsorContainer container, IConfigurationStore store)

{

container.Register(Component.For<MappingConfiguratorBase>().ImplementedBy<Scanning2CoreMappersConfiguration>().LifestyleSingleton());

container.Register(Component.For<IMapper<Medium, MediumDataContract>>().ImplementedBy<SimpleMapper<Medium, MediumDataContract>>().LifestyleSingleton());

container.Register(Component.For<IMapper<PartName, PartNameDataContract>>().ImplementedBy<SimpleMapper<PartName, PartNameDataContract>>().LifestyleSingleton());

}

}

}

**Problem :**  Namapovana tabulka neni zaregistrovana v Castlu ( ServAction has dependencies on IenumTableValues which was not registered ) **Reseni : Na mapovane tabulce chybel atribut jake je to dao : Napr :**

[DaoFactory(DaoType = typeof(TempMotivletDao))]