# Jak pomoci CastleWindsor resolvovat genericke tridy na zaklade generickeho argumentu.

Ve zmenovadle jse potreboval tridu ktera by mi insertla do databaze **ValueDataContract (VDC)** na zaklade toho o jaky VDC se jedna . VDC je totiz abstraktni trida od ktere dedi napr OwnerDataContract, ProductBrandDataContract atd.

* Vytvorim si rozhrani ktere budou implementovat vsechny moje insertery. Rozhrani je nutne, protoze metoda CreateInserter na factory tride vraci Iinserter.

public interface IInserter

{

DoneResponse Insert<T>(T obj) where T: ValueDataContract;

}

* Nyni si udelam deset trid (inserteru) ktere budu pouzivat na insertovani do ruznych dao trid

public class CategoryInserter: AuditableInserterBase, IInserter

{

private readonly CategoryDao m\_categoryDao;

public CategoryInserter(IAuditableIdentityProvider provider, CategoryDao categoryDao) : base(provider)

{

m\_categoryDao = categoryDao;

}

public DoneResponse Insert<T>(T obj) where T : ValueDataContract

{

throw new System.**NotImplementedException**();

}

}

* Vytvorim Factory tridu ktera mi pomoci Castle resolvne vzdy jinou tridu insertera podle generickeho argumentu.

public class Factory

{

private readonly IWindsorContainer m\_container;

public Factory(IWindsorContainer container)

{

m\_container = container;

}

public IInserter CreateInserter<T>() where T : ValueDataContract

{

if (typeof(T) == typeof(OwnerDataContract)) **// pokud je genericky argument OwnerDataContract**

{

return m\_container.Resolve<OwnerInserter>(); **// vrat OwnerInsertera**

}

if (typeof(T) == typeof(CompanyBrandDataContract))

{

return m\_container.Resolve<CompanyBrandInserter>(); **// vrat CompanyInsertera**

}

return null;

}

}

* Instance Factory se vytvari v nejake servisni akci kde chci insertovat pokazde do jineho Daa. M\_container v servisni akci ziskam tak, ze ho proste dam jako zavislost konstruktoru serv.akce. Pak ho predam konstruktoru Factory tridy. Neni mozne si ve factory vytvorit novou instanci windsorContaineru, potrebuju tu jiz vytvorenou.

var factory = new Factory(m\_container);

var inserter = factory.CreateInserter**<OwnerDataContract**>();

var response = inserter.Insert(new OwnerDataContract());

var anotherInserter = factory.CreateInserter<CompanyBrandDataContract>();

var resp = anotherInserter.Insert(new CompanyBrandDataContract());

Aby Castle mohl vytahovat (resolvovat) tridy z conteineru, musim je tam nejdrive zaregistrovat. Udelam to tam kde registruju ViewModely tedy v ChangingInstalleru:

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

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

container.Register(Component.For<CompanyBrandInserter>().ImplementedBy<CompanyBrandInserter>().LifestyleSingleton()); ATD.

**V servisni akci vlasten rikam : Je genericky argument factory metody OwnerDataContract ? Vrat mi insertera ktery bude insertovat do OwnerDao. Je to jiny typ ValueDataContractu ? Vrat mi insertera podle tohoto typu. Ktereho insertera dostanu rozlisuje factory metoda CreateInserter pomoci ifu na zaklade typu.**

# Factory ktera vraci IfilterElement podle zadaneho generickeho argumentu

* **Ve zmenovadle mame datacontracty napr:**

OwnerDataContract: ValueDataContract

ProductBrandDataContract : ValueDataContract

Atd..

* **Mame tu generickou tridu ktera implementuje rozhrani IFilterElement:**

public class FilterElement<T> : PropertyChangedBase, IFilterElement where T : ValueDataContract, new()

* **Instanci FilterElementu musim tvorit pomoci zadani generickeho argumentu kterym je datacontract jenz dedi od ValueDataContractu:**

new FilterElement<MotiveNameDataContract>()

**Ve zmenovadle na zalozce prejmenovani potrebuju vytvorit instanci IfilterElement podle toho, jaky datacontract zvoli uzivatel v hornim comboboxu kde items je kolekce datacontractu.**

Vytvorim teda generickou tridu IfilteElementFactory kde T je ValueDataContract a klicove slovo new() mi dovoli vytvorit instanci tohoto typu.:

public class IFilteElementFactory <T> where T : ValueDataContract, new()

{

public static IFilterElement CreateIFilterElement ()

{

return new FilterElement<T>();

}

}

Vraci sice FilterElement ale to je zaroven I rozhrani IFilterElement.

**Instanci tohoto rozhrani si vyrobime takto :**

var test = IFilteElementFactory<MotiveNameDataContract>.CreateIFilterElement();

var neco = test.GetType(); // FilterElement<MotiveNameDataContract>

var neco1 = test.GetType().GetGenericElementType(); // MotiveNameDataContract

Jaky genericky argument factory predhodime, takovy IFilterElement dostaneme.

# Factory if - return

public class IFilteElementFactory

{

public static IFilterElement CreateIFilterElement(IClientToServicePublisher publisher, ValueDataContract vdc)

{

if (vdc.GetType().IsAssignableFrom(typeof(OwnerDataContract)))

{

return new FilterElement<OwnerDataContract>(publisher);

}

if (vdc.GetType().IsAssignableFrom(typeof(CompanyBrandDataContract)))

{

return new FilterElement<CompanyBrandDataContract>(publisher);

}

...

return null;

}

}

Ale kdyz jsme u toho, dala by se ta factory resit jinak ?

## [‎5/‎21/‎2018 9:45 AM] Filip Čálek:

no nevim. blbe. kdyby se ty filterElementy resolvovaly z containeru, tak by to bylo jednodussi

[‎5/‎21/‎2018 9:46 AM] Peter Hlavenka:

to ano, ale i tak by se mu muselo neco strcit jako argument

[‎5/‎21/‎2018 9:46 AM] Filip Čálek:

to se pak pouziva bezne. container.Resolve(typeof(vdc))

[‎5/‎21/‎2018 9:46 AM] Peter Hlavenka:

aha a to vdc muze byt tady variable ?

[‎5/‎21/‎2018 9:47 AM] Filip Čálek:

vdc = ValueDataContract

[‎5/‎21/‎2018 9:47 AM] Peter Hlavenka:

to jo, me ale psalo ze parametr metody neni type ale variable a to jako argument pouzit nejde

[‎5/‎21/‎2018 9:47 AM] Filip Čálek:

no ale on pravdepodobne uvnitr ma pak stejne podminku if(vdc.getype()...) atd. >]

[‎5/‎21/‎2018 9:47 AM] Peter Hlavenka:

ok dekuju

[‎5/‎21/‎2018 9:48 AM] Filip Čálek:

no vsak je. ale container umi resolvovat pres typ jako genericky argument a nebo explicitne

Resolve<Type>

nebo Resolve(Type neco)

ale to jsem se dostali nekam dal:) nas pripad to neresi;)

# Typed Factory Facility - interface-based factories

<https://github.com/castleproject/Windsor/blob/master/docs/typed-factory-facility-interface-based.md>

1. Vytvorime rozhrani

public interface IValueDatacontractFactory

{

IValueDataContract Create();

void Release(IValueDataContract component);

OwnerDataContract GetOwnerDataContract();

}

1. Do containeru zaregistrujeme Factory

container.AddFacility<TypedFactoryFacility>();

container.Register(Component.For<IValueDatacontractFactory>().AsFactory());

1. Zaregistrujeme vsechna rozhrani ktere budu chtit resolvovat

container.Register(Component.For<IValueDataContract>().ImplementedBy<OwnerDataContract>().Named("OwnerDataContract").LifestyleTransient());

1. Kdyz ma componenta pridane jmeno , na Factory rozhrani si staci dat metodu Get+jmeno , ktera vraci pozadovanou instanci .
2. Instanci si vytvorime tak, ze si do konstruktoru pridame dependency na Factory

public RenamingViewModel(DataViewModel dataViewModel, IClientToServicePublisher publisher, IValueDatacontractFactory factory)

{

m\_factory = factory;

}

1. A zavolame:

var ownerDataContract = m\_factory.GetOwnerDataContract();

**Aby bylo mozne resolvnout Castlem tridu, resolvovana trida musi mit bezparametricky konstruktor.**

# Resovnuti typu z Factory , bez nutnosti resolvovat samotnou Factory

[‎18.‎09.‎2018 9:02]  Peter Hlavenka:

            container.Register(Component.For<InstitutionalSectorToOwnershipCacheFactory>().LifestyleSingleton().Named("institutionalSectorToOwnershipCacheFactory")

                .DependsOn(

                    Property.ForKey("serializedCachePath").Eq(configurationProvider.GetConfig<AddressConfiguration>().SerializedCachePath)));

            container.Register(Component.For<InstitutionalSectorToOwnershipCache>().LifestyleSingleton()

                .DependsOn(

                    Property.ForKey("InternalCache").Eq(container.Resolve<InstitutionalSectorToOwnershipCacheFactory>().Create().InternalCache)));

[‎18.‎09.‎2018 9:05]  Karel Honzl:

zajmave

[‎18.‎09.‎2018 9:06]  Peter Hlavenka:

mozna to jde i lip, tohle je moje reseni

[‎18.‎09.‎2018 9:08]  Karel Honzl:

v jake solutione to najdu?

[‎18.‎09.‎2018 9:09]  Peter Hlavenka:

zatim jen v moji vetvi Kodovadla d55e801f55ef7165705c67f5c6afe11d6eaa70a1

jinak ta factory je v Mediaresearch.Framework.Utilities.Ares

Tuto konverzaci jsme uložili. Brzo ji najdete na kartě Konverzace ve Skypu pro firmy a ve složce Historie konverzací v Outlooku.

[‎18.‎09.‎2018 9:47]  Peter Hlavenka:

**Nasel jsem factory jeste v MIR.Media.Coding.Container.Installers   a mozna lip :**  
  
                     container.Register(

                           Component.For<XmlPathInfoProviderFactory>()

                                  .DependsOn(

                                         Property.ForKey("assemblyName").Eq("MIR.Media.Coding"),

                                         Property.ForKey("resourcePath").Eq("Authentication.PathInfos.xml")

                                  ),

                           Component.For<IPathInfoProvider>()

                                  .UsingFactoryMethod(kernel => kernel.Resolve<XmlPathInfoProviderFactory>().Create()).Named("pathInfoProvider")

                           );