Bootstrapper

##### Kod bootstrapperu

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Globalization;

using System.IO;

using System.Reflection;

using System.Security.AccessControl;

using System.Security.Principal;

using System.Threading;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Markup;

using Caliburn.Micro;

using Castle.DynamicProxy;

using Castle.MicroKernel.Registration;

using Castle.Windsor;

using Castle.Windsor.Configuration.Interpreters;

using Castle.Windsor.Installer;

using log4net.Config;

using Mediaresearch.Framework.Communication.Common;

using Mediaresearch.Framework.DataAccess.BLToolkit.Dao;

using Mediaresearch.Framework.DataAccess.Core.Auditable;

using Mediaresearch.Framework.Mapping;

using MIR.Common2.Gui;

using MIR.PrintStorage.Scanning2.Common;

using MIR.PrintStorage.Scanning2.Common.Auditable;

using MIR.PrintStorage.Scanning2.Common.Scan;

using MIR.PrintStorage.Scanning2.Core;

using MIR.PrintStorage.Scanning2.GUI.ViewModels;

using MIR.PrintStorage.Scanning2.Shell.Installers;

using MIR.PrintStorage.Scanning2.Shell.Properties;

using Action = System.Action;

using GuiResources = MIR.PrintStorage.Scanning2.GUI.GuiResources;

using ILog = log4net.ILog;

using LogManager = log4net.LogManager;

namespace MIR.PrintStorage.Scanning2.Shell.Bootstrapper

{

public class ScanningBootstrapper : Bootstrapper<MainViewModel>

{

private static readonly string m\_applicationDirectory = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);

private static readonly ILog m\_log = LogManager.GetLogger(MethodBase.GetCurrentMethod().DeclaringType);

private WindsorContainer m\_globalContainer;

protected override void Configure() // metoda caliburnu viz <https://d.docs.live.net/b22fb0fb09218bf0/Nielsen%20%20prace/Moje%20poznamky%20Nielsen/Caliburn/Start%20From%20Bootstrapper%20using%20Caliburn.docx>

{

ViewLocator.LocateForModel = (model, displayLocation, context) =>

{

var unproxiedModelType = ProxyUtil.GetUnproxiedType(model);

return ViewLocator.LocateForModelType(unproxiedModelType ?? model.GetType(), displayLocation, context);

};

}

protected override void OnStartup(object sender, StartupEventArgs e)

{

ConfigureLog4Net();

ConfigureLanguage();

m\_globalContainer = new WindsorContainer(new XmlInterpreter(Path.Combine(m\_applicationDirectory, "MIR.PrintStorage.Scanning2.Shell.Container.config"))); **// definuje xml soubor z ktereho se nacitaji property pro scannigConfiguration viz mojePoznamky => ContainerConfig**

var configuration = m\_globalContainer.Resolve<PrintStorageDbConfiguration>(); // resolvne z xml-ka tridu PrintStorageDbConfiguration

IScannigConfiguration scanningConfiguration = m\_globalContainer.Resolve<IScannigConfiguration>(); // resolvne z xml tridu scanningConfiguration a vlozi ji do promenne jako rozhrani

string path = scanningConfiguration.IconPath;

IWindowManager windowManager = new WindowManager();

m\_globalContainer.Register(Component.For<IWindowManager>().Instance(windowManager));

MainViewModel rootViewModel = null;

var assembly = Assembly.GetExecutingAssembly();

FileVersionInfo fvi = FileVersionInfo.GetVersionInfo(assembly.Location);

string version = $"{GuiResources.Scanning} {fvi.FileVersion}";

string appVersionInfo = $"{version} ({configuration.DefaultDataSource})";

Action initAction = () => // Tato cast se pusti asynchronne, mezitim se zobrazuje SplashScreen. Ten dostane tuto akci jako parametr konstruktoru. Tim vi kdy akce skonci a zavre se az po skonceni provadeni

{

m\_globalContainer.Register(Component.For<IWindsorContainer>().Instance(m\_globalContainer).LifestyleSingleton());

m\_globalContainer.Install(FromAssembly.This(), FromAssembly.Containing<ScanningCoreAssemblyIdentificator>()); //Spusti metody Install ze vsech Instalatoru

m\_globalContainer.Resolve<IMediaDataDaoSource>();

m\_globalContainer.Resolve<IPrintStorageDaoSource>();

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

mappingConfiguratior.Configure();

if (!Directory.Exists(ScanningConfiguration.TempFilesDirectory))

{

Directory.CreateDirectory(ScanningConfiguration.TempFilesDirectory); // Ve tride ScanningConfiguration (kterou si castle bere rovnou z xml souboru) je natvrdo definovano: public static string TempFilesDirectory { get; } = Path.Combine(Path.GetTempPath(), "Scanning");

}

ConfigureSubscriber();

ConfigureUser();

rootViewModel = m\_globalContainer.Resolve<MainViewModel>();

rootViewModel.AppVersionDescription = appVersionInfo;

};

SplashScreenViewModel splashScreen = new SplashScreenViewModel(initAction, version, path);

windowManager.ShowDialog(splashScreen); // Zobrazi se splash, ktery si vnitrne spusti initAction

windowManager.ShowDialog(rootViewModel); // Po skonceni splashe se spusti aplikace coz je taky dialog

Application.Shutdown();

m\_globalContainer.Dispose();

}

private void ConfigureUser() // Prihlasovani uzivatele

{

var userSource = m\_globalContainer.Resolve<IUserSource>(); // Kdyz Castlu rekneme, ze chceme rozhrani tak vytvori instanci tridy ktera toto rozhrani implementuje (vi podle registrace)

var auditableIdentityProvider = m\_globalContainer.Resolve<IAuditableIdentityProvider>();

var login = WindowsIdentity.GetCurrent().Name;

if (userSource.Login(login)) // UserSource je jednoducha trida, ktera se prez UserDao podiva do databaze, jestli tam existuje uzivatel login, coz je WindowsIdentity.GetCurrent().Name

{

var userId = userSource.GetUserId();

auditableIdentityProvider.SetAuditableIdentity(new AuditableIdentity(userId, login));

return true;

}

var message = new MessageBoxViewModel(GuiResources.UserNotFoundError, false, 3000); // Pokud uzivatele nenajde v databazi, tak nema opravneni spustit aplikaci, zobrazi se dialogove okno a ukonci se aplikace

windowManager.ShowDialog(message);

return false;

}

private static void ConfigureLog4Net() // Zapisovani do Logu viz <https://d.docs.live.net/b22fb0fb09218bf0/Nielsen%20%20prace/Moje%20poznamky%20Nielsen/Log.docx>

{

var logConfigFile = new FileInfo(Path.Combine(m\_applicationDirectory, "MIR.PrintStorage.Scanning2.Shell.log4net"));

XmlConfigurator.Configure(logConfigFile);

AppDomain.CurrentDomain.UnhandledException += CurrentDomainUnhandledException;

}

private void ConfigureLanguage()

{

var cultureInfo = CultureInfo.GetCultureInfo(Settings.Default.Language);

FrameworkElement.LanguageProperty.OverrideMetadata(typeof(FrameworkElement), new FrameworkPropertyMetadata(XmlLanguage.GetLanguage(cultureInfo.Name)));

Thread.CurrentThread.CurrentCulture = cultureInfo;

Thread.CurrentThread.CurrentUICulture = cultureInfo;

GuiResources.Culture = cultureInfo;

}

// kde se maji hledat requesty a responsy

private void ConfigureSubscriber()

{

var coreAssembly = typeof(ScanningCommonAssemblyIdentificator).Assembly;

var notificationProvider = m\_globalContainer.Resolve<INotificationsReceiversAssemblyProvider>();

notificationProvider.RegisterAssemblies(coreAssembly, coreAssembly);

var notificationsSubcriber = m\_globalContainer.Resolve<INotificationReceiverSubscriber>();

notificationsSubcriber.SubscribeAll();

var provider = m\_globalContainer.Resolve<IRequestsServiceActionsAssemblyProvider>();

provider.RegisterAssemblies(coreAssembly, typeof(ScanningCoreAssemblyIdentificator).Assembly);

var subscriber = m\_globalContainer.Resolve<IServiceActionSubscriber>();

subscriber.SubscribeAll();

}

// kde se maji hledat view

protected override IEnumerable<Assembly> SelectAssemblies()

{

return new[] {Assembly.GetAssembly(typeof(MainViewModel)), Assembly.GetAssembly(typeof(Common2GuiAssembyIdentificator))};

}

private static void CurrentDomainUnhandledException(object sender, UnhandledExceptionEventArgs e)

{

Log(e.IsTerminating, e.ExceptionObject, m\_log);

}

// zapise vyjimku do Logu

public static void Log(bool isTerminating, object exceptionObject, ILog logger)

{

var message = $"Unhandled exception in application (IsTerminating = {isTerminating})";

/\*

\* Why is UnhandledExceptionEventArgs.ExceptionObject of type Object and not Exception?

\* While not all languages support throwing non-Exception type exceptions, the CLR and IL allow for throwing any Object.

\* In general, throwing non-Exception types is discouraged because most developers do not expect this to occur,

\* and are not likely to catch the object.

\* On the other hand, developers who are overriding the unhandled exception logic may need to catch non-Exception objects, also.

\*/

if (exceptionObject is Exception ex)

{

logger.Fatal(message, ex);

}

else

{

logger.Fatal($"{message} : {exceptionObject}");

}

}

}

}