Tvorba pluginu do IS Helios v MS Visual Studio

Tento dokument podrobně popisuje postup pro tvorbu pluginu do ERP Helios (Orange) v MS Visual Studio (C# 2010 Express), jeho následnou registraci na klientském počítači a konečné použití v ERP Helios.

Obsah

[1 Předpoklady 2](#_Toc324532888)

[2 Příprava Interface pro práci v MS Visual Studio 2](#_Toc324532889)

[3 Tvorba pluginu v MS Visual Studio C# 2010 Express 2](#_Toc324532890)

[3.1 Přidaní Interface (ddPlugin.dll) do projektu 4](#_Toc324532891)

[3.2 Viditelnost knihovny pro COM aware aplikace 5](#_Toc324532892)

[3.3 Tvorba pluginu 6](#_Toc324532893)

[3.3.1 Tvorba třídy 6](#_Toc324532894)

[3.3.2 Zadání nezbytných informací 8](#_Toc324532895)

[3.3.3 Tvorba funkcionality pluginu 9](#_Toc324532896)

[3.4 Kompilace knihovny s pluginem 13](#_Toc324532897)

[4 Registrace knihovny s pluginem 13](#_Toc324532898)

[5 Použití pluginu v IS Helios 15](#_Toc324532899)

# Předpoklady

Základními předpoklady pro tvorbu pluginu jsou:

* Soubor ddPlugin.tlb – na vývojovém prostředí nezávislý popis Interface pro komunikaci s prostředím IS Helios.
* Sériové číslo od Asseco Solutions na vývoj pluginu
* Zaregistrování jména knihovny pluginu u Asseco Solutions
* Vývojové prostředí MS Visual Studio včetně SDK

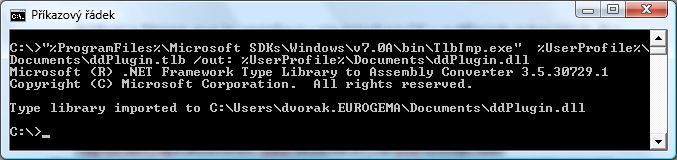
# Příprava Interface pro práci v MS Visual Studio

Předtím než začneme se samostatným programováním, je nutné vytvořit nativní překlad Interface (COM rozhraní) pro používanou platformu .NET. Toto se provádí voláním aplikace TlbImp.exe z Windows SDK nad souborem ddPlugin.tlb.

Aplikace TlbImp.exe je umístěna v závislosti na OS v odlišných adresářích. Pro Windows 7 je tato cesta %ProgramFiles%\Microsoft SDKs\Windows\v7.0A\bin\.

Přípravou se rozumí tvorba dll knihovny a to následným způsobem (ddPlugin.tlb byl nakopírován do %UserProfile%\Documents):

"%ProgramFiles%\Microsoft SDKs\Windows\v7.0A\bin\TlbImp.exe" %UserProfile%\Documents\ddPlugin.tlb /out: %UserProfile%\Documents\ddPlugin.dll

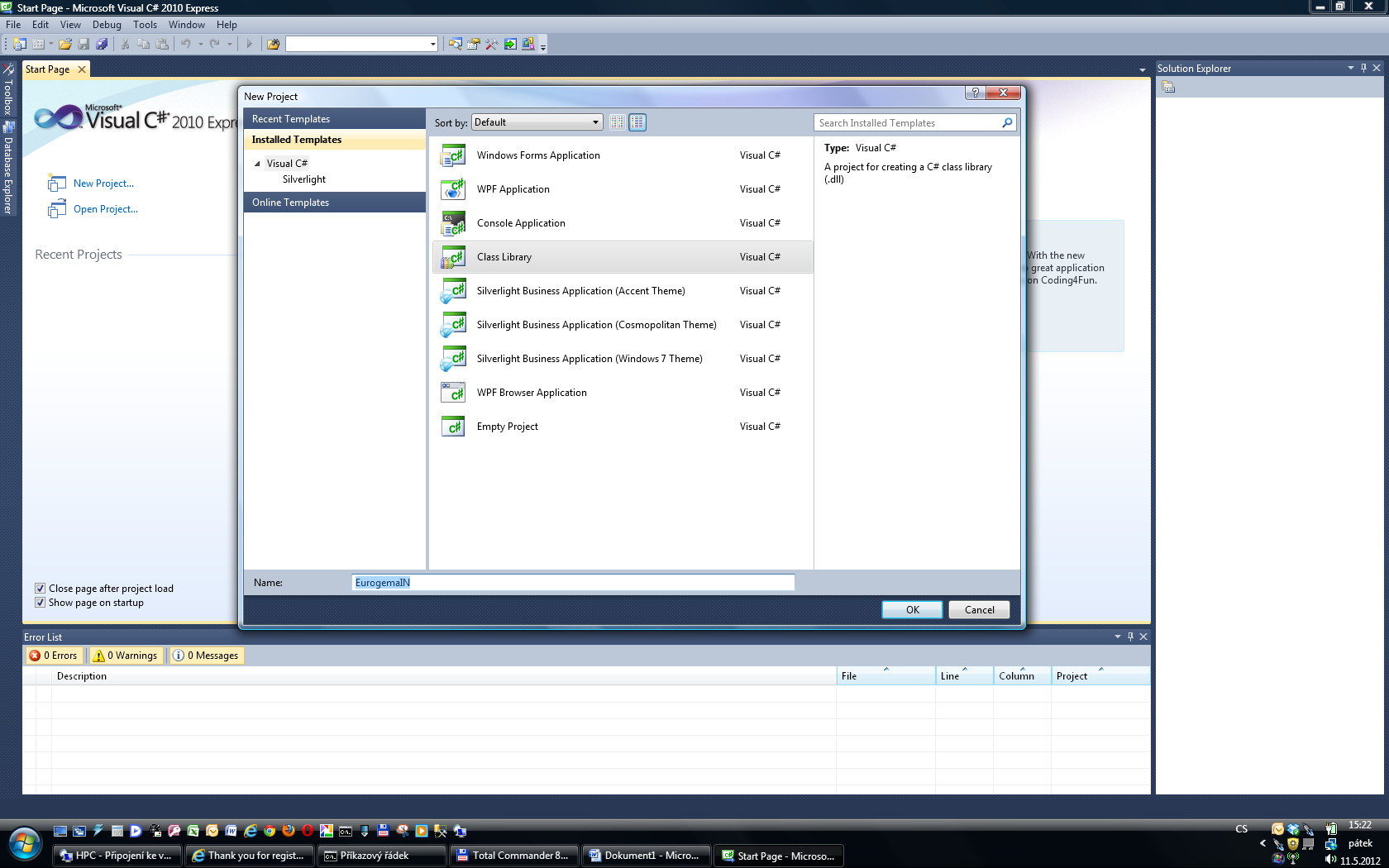


Celý výše uvedený příkaz je na jedné řádce. Výsledný soubor je umístěn na cestě definované pomocí přepínače /out:. V našem případě tedy na %UserProfile%\Documents\ddPlugin.dll.

[Soubory ddPlugin.tlb a ddPlugin.dll jsou uloženy v adresáři Interface. Je zde také uložen opis webové stránky <https://forum.helios.eu/orange/doc/cs/Helios_Orange_Interface>, který obsahuje popis jednotlivých metod rozhraní.]

# Tvorba pluginu v MS Visual Studio C# 2010 Express

Po spuštění Visual Studia vybereme akci „New Project …“ a jako typ aplikace zvolíme „Class Library“. Pod doplnění názvu projektu (v našem případě EurogemaIN – toto bude jméno knihovny pluginu, které se musí zaregistrovat na stránkách AssecoSolutions) stiskneme OK.

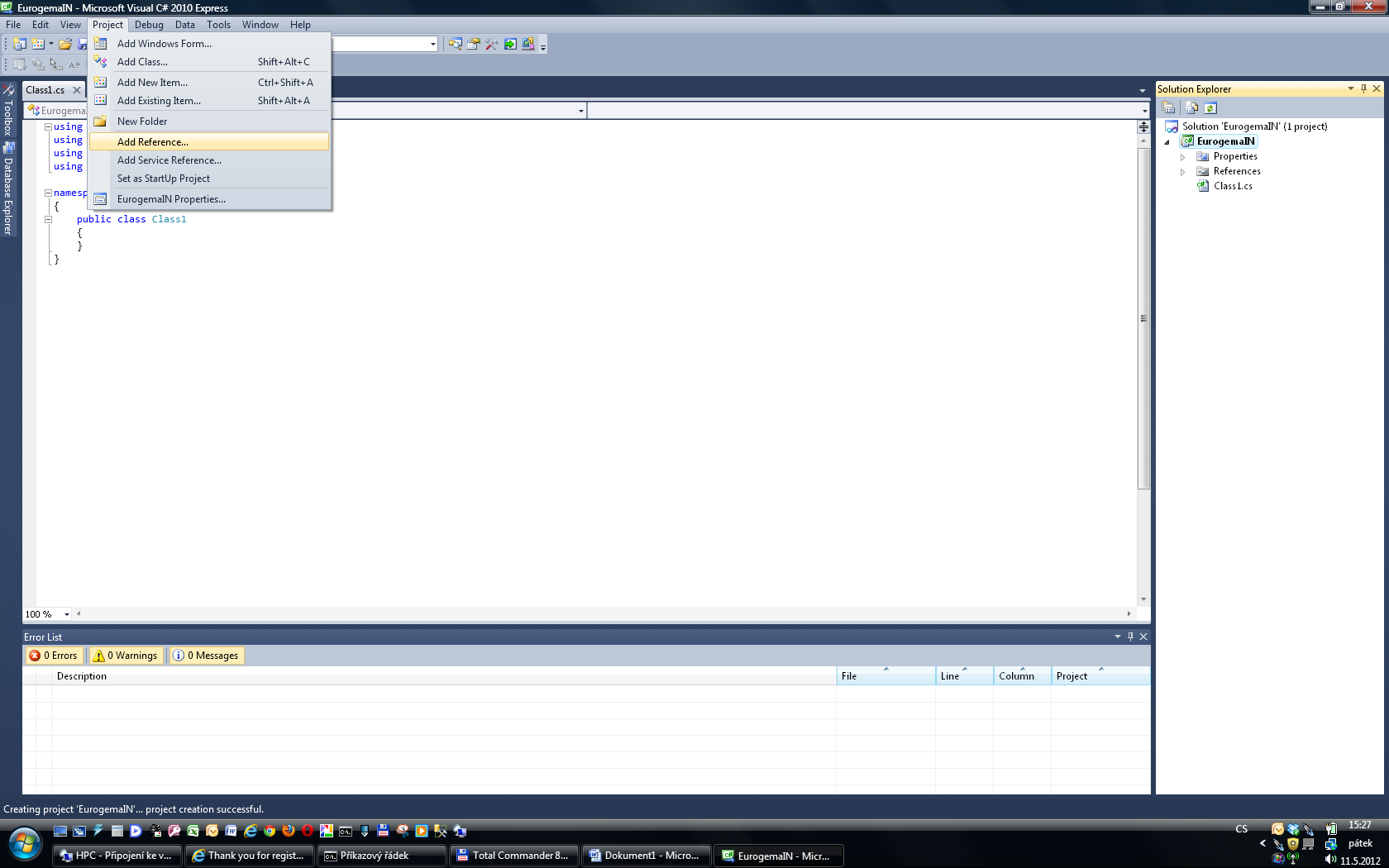


Ještě než začneme se samotnou tvorbou pluginu, je třeba provést následné operace.

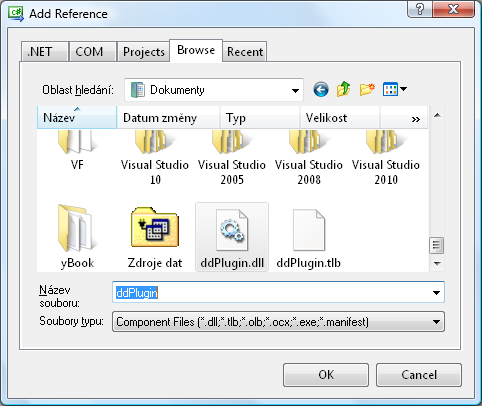
## Přidaní Interface (ddPlugin.dll) do projektu

Nejprve je nutno do projektu přidat referenci na knihovnu s Interface Helios. Toto je soubor ddPlugin.dll, který jsme si vytvořili v bodě 2 tohoto dokumentu.

Akci provedeme zvolením akce „Add Reference..“ z menu „Project“.



Poté na záložce „Browse“ nalezneme soubor ddPlugin.dll a stiskneme OK.

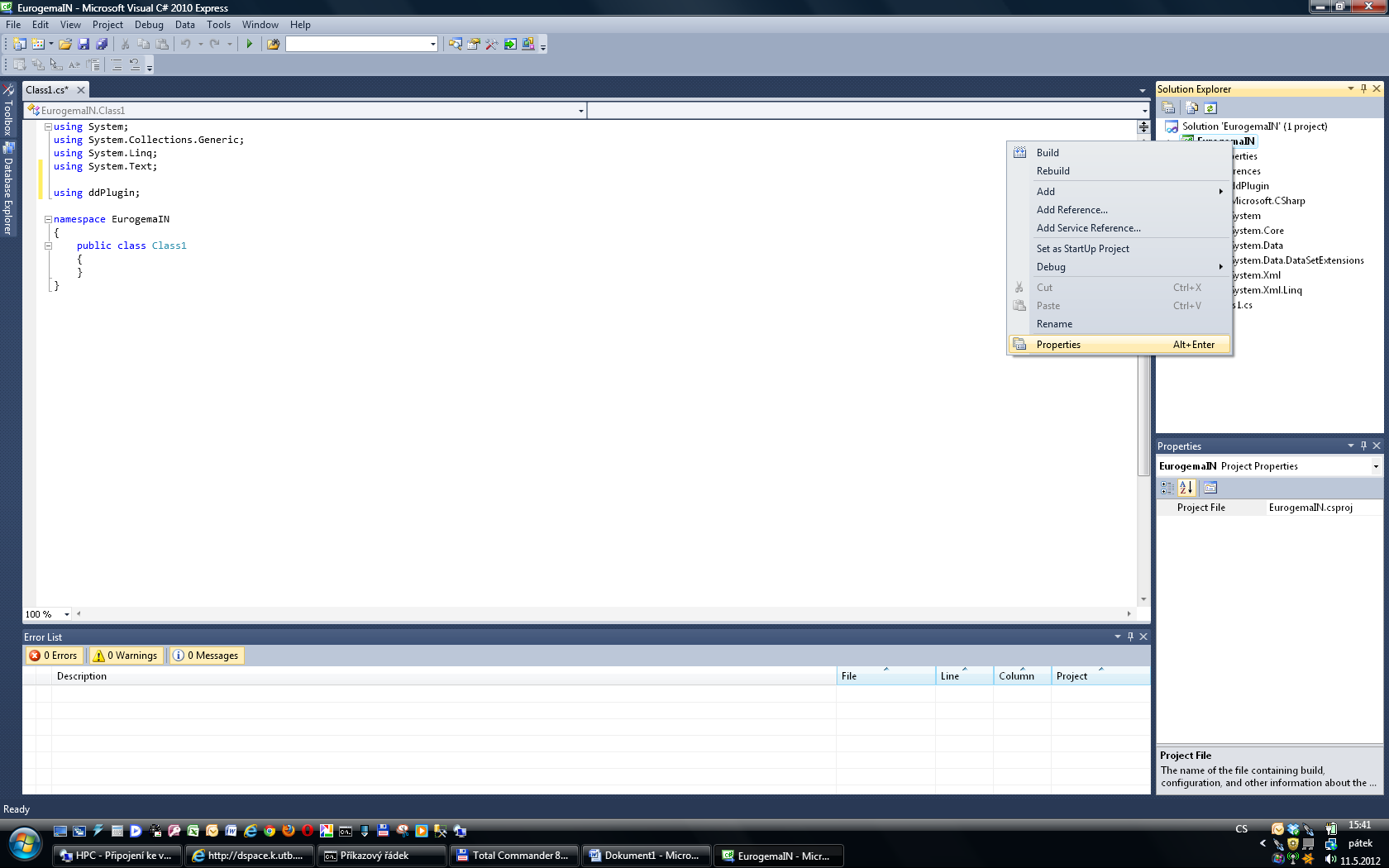


Tím jsme si vytvořili možnost volat z kódu funkce Interface Helios.

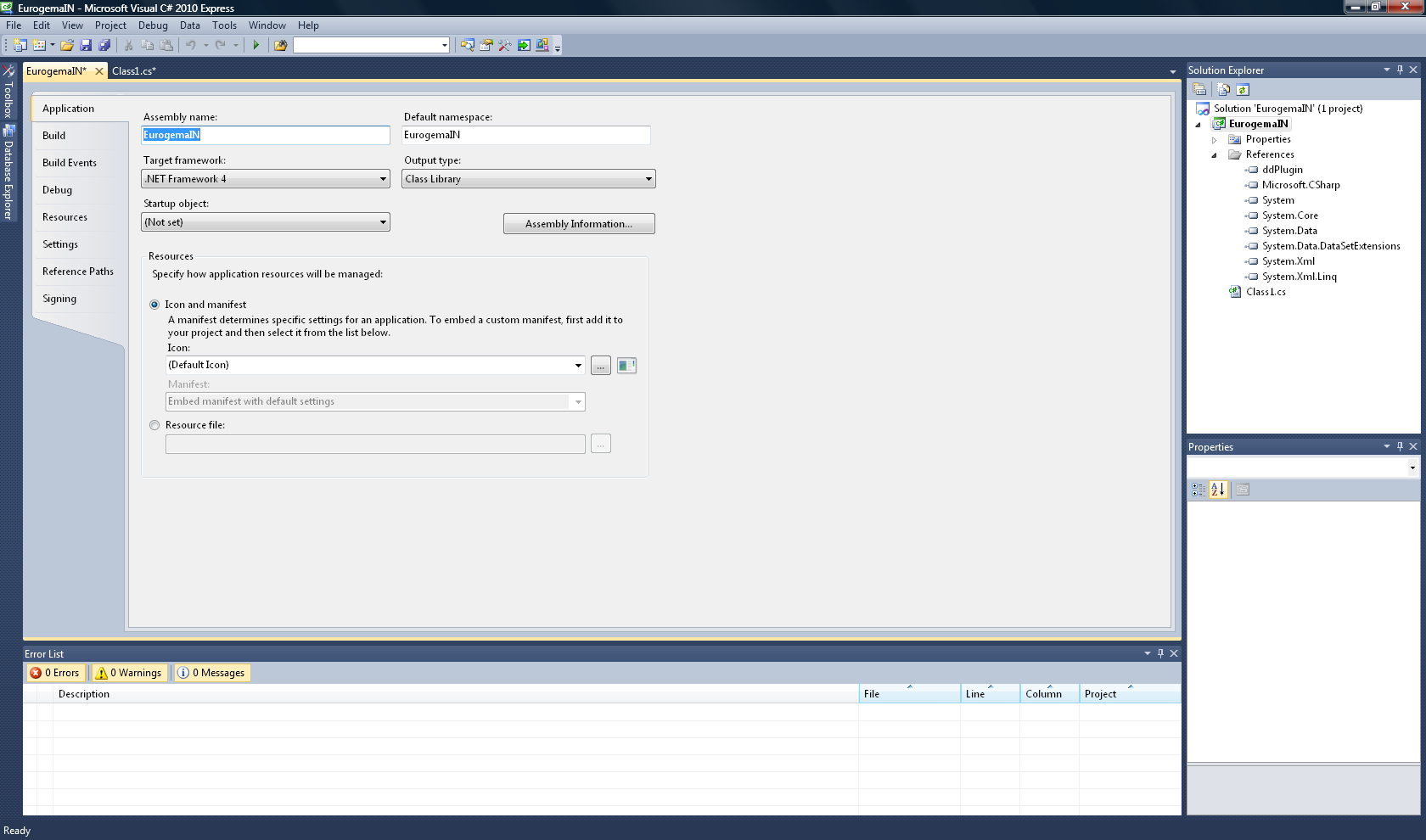
## Viditelnost knihovny pro COM aware aplikace

Dalším důležitým krokem je nastavení projektu tak, aby generoval knihovnu jako COM Server, což je podmínka funkčnosti pluginu v systému Helios.

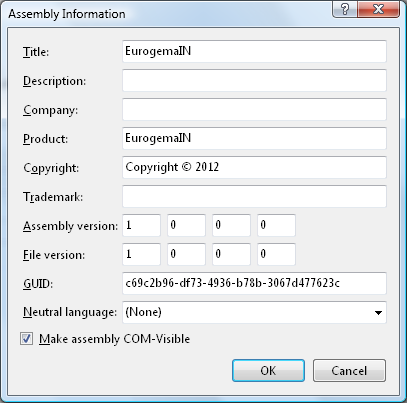
Toto provedeme kliknutím pravým tlačítkem na projekt (EurogemaIN) v Solution Explorer a zvolením akce „Properties“.



Na listu Application klikneme na tlačítko „Assembly Information…“ (Za pozornost stojí výběr Target framework. Určuje způsob, jakým budeme později pluginy registrovat).



A v následném dialogu zaškrtneme volbu „Make assembly COM-Visible“ a stiskneme OK.



Nyní již nic nebrání tomu, aby se jednotlivé třídy projektu mohly stát externími akcemi typu Plugin v IS Helios.

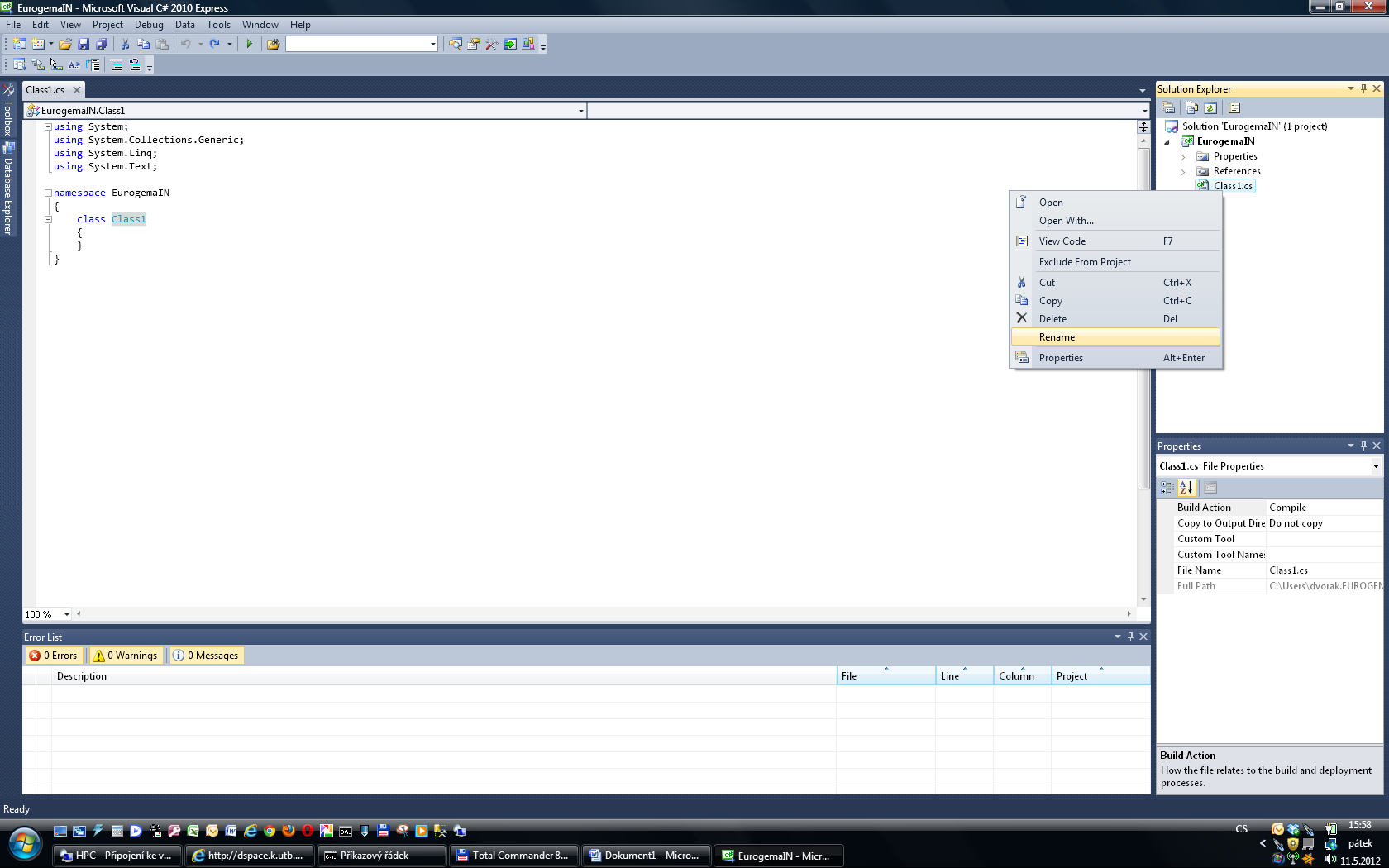
## Tvorba pluginu

### Tvorba třídy

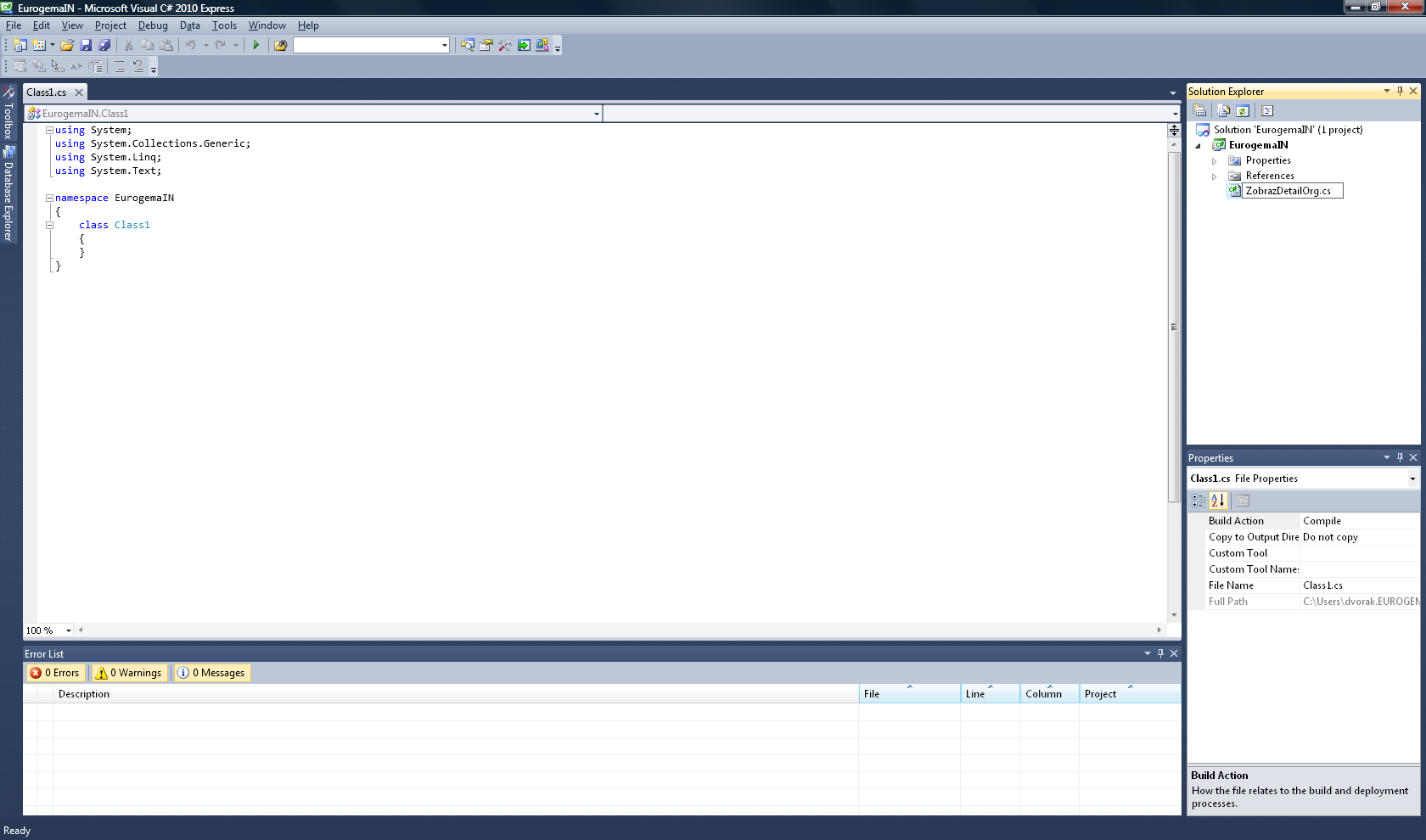
Samotná tvorba pluginu je vlastně vytvoření veřejné třídy v jemném prostoru názvu pluginu.

Námi definovaná třída bude mít velmi jednoduchou funkcionalitu a to že nad vybranou organizací zobrazí v dialogu její jméno a IČO. Pojmenujeme ji ZobrazDetailOrg.

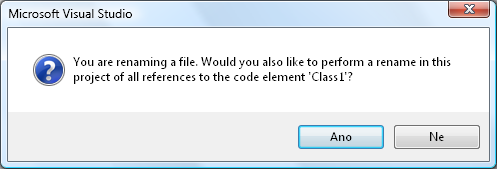
Nejprve přejmenujeme existující třídu Class1 na ZobrazDetailOrg tak, že v Solution Exploreru klikneme pravý tlačítkem na soubor Class1.cs a vybereme možnost Rename.



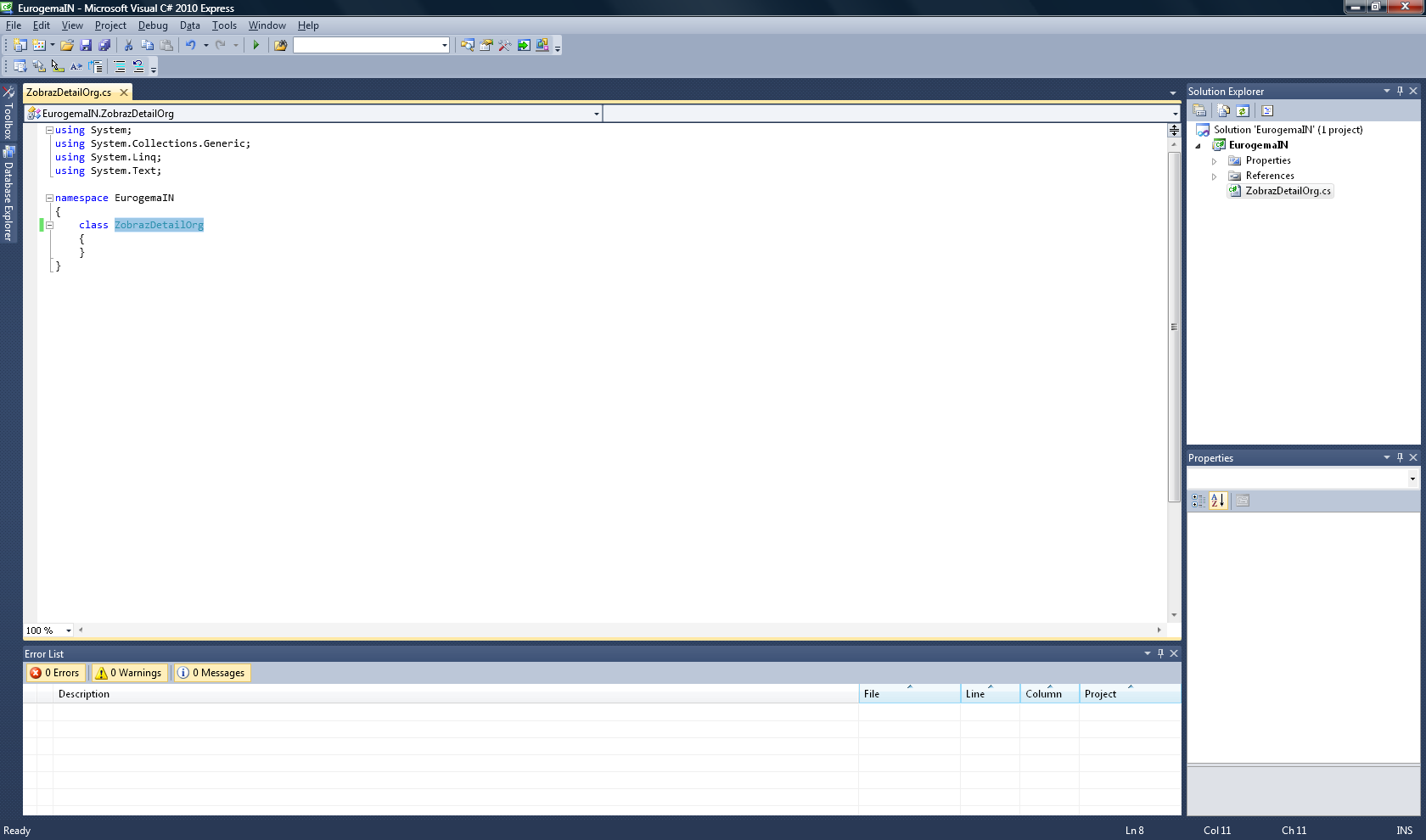
Následně nazveme soubor ZobrazDetailOrg.cs.



V následném dialogu odklikneme možnost Ano.



Tím jsme si nejen přejmenovali soubor, ale také třídu.



Všimněme si, jak je strukturován jmenný prostor. V našem případě je třída ZobrazDetailOrg ve jmenném prostoru EurogemaIN. Jmený prostor EurogemaIN musí odpovídat názvu dll knihovny s pluginy. Navíc musí být název jmenného prostoru zaregistrován na stránkách společnosti Asseco Solutions pod sériovým číslem vývojáře! Název třídy ZobrazDetailOrg pak odpovídá jménu pluginu. V části 5 (Použití pluginu v IS Helios)se k tomu ještě vrátíme.

Nyní bychom měli celý projekt s veškerými nastaveními uložit.

### Zadání nezbytných informací

Pro přehlednost kódu je doporučeno zadat odkaz na knihovnu s Interface Helios (ddPlugin.dll) do klauzule using.

Třída musí být public, aby ji Helios viděl.

Dále je nutno zaručit, aby námi vytvořená knihovna implementovala rozhraní IHePlugin2.

V neposlední řadě je nutno implementovat dvě metody:

* PartnerIdentification – tato metoda by měla Helios Orange vrátit identifikaci (sériové číslo Helios Orange) tvůrce pluginu
* Run – samotné tělo pluginu, jemuž je předán odkaz na rozhraní IHelios.

Definice třídy vypadá v této chvíli následně:

using System;

using ddPlugin;

namespace EurogemaIN

{

public class ZobrazDetailOrg: IHePlugin2

{

public string PartnerIdentification()

{

return @"<zde patří sériové číslo vývojáře>";

}

public void Run(IHelios Helios)

{

}

}

}

Po uložení a zkompilování projektu je již možno plugin zaregistrovat a začít používat. V této podobě však nebude plugin provádět žádnou práci.

### Tvorba funkcionality pluginu

Hned zpočátku si ustanovme premisu, že v našem ukázkovém pluginu nebudeme provádět kontroly na různé podmínky při jeho používání. V ostré verzi na ně ovšem nesmíme zapomenout.

Proces funkcionality můžeme rozložit do následných korků:

1. Zjištění ID záznamu, nad kterým uživatel provádí externí akci.
2. Tvorba formuláře pro zobrazení detailů.
3. Zobrazení informací ve formuláři.

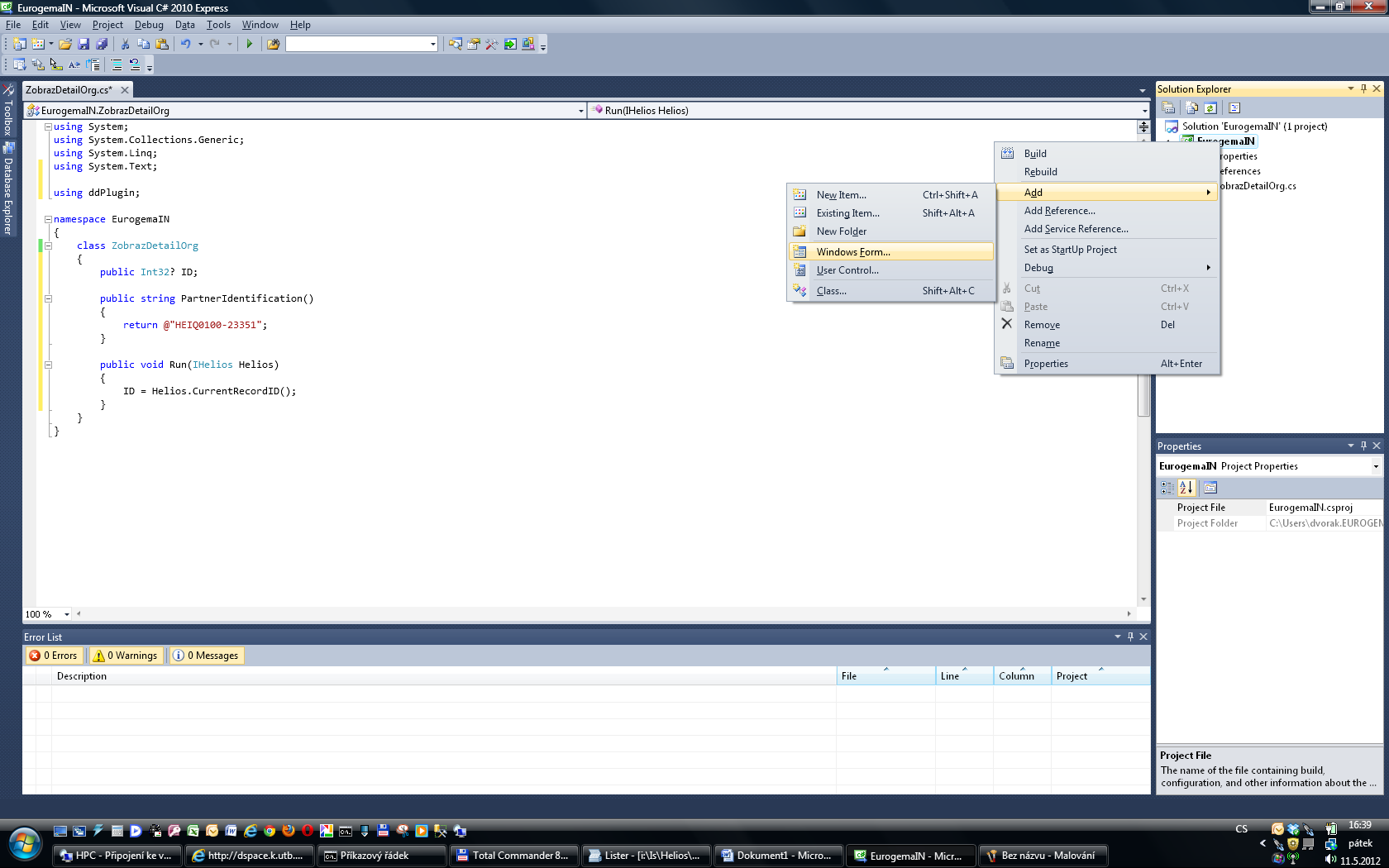
#### Zjištění ID záznamu

Nadefinujeme si proměnou ID typu Int32.

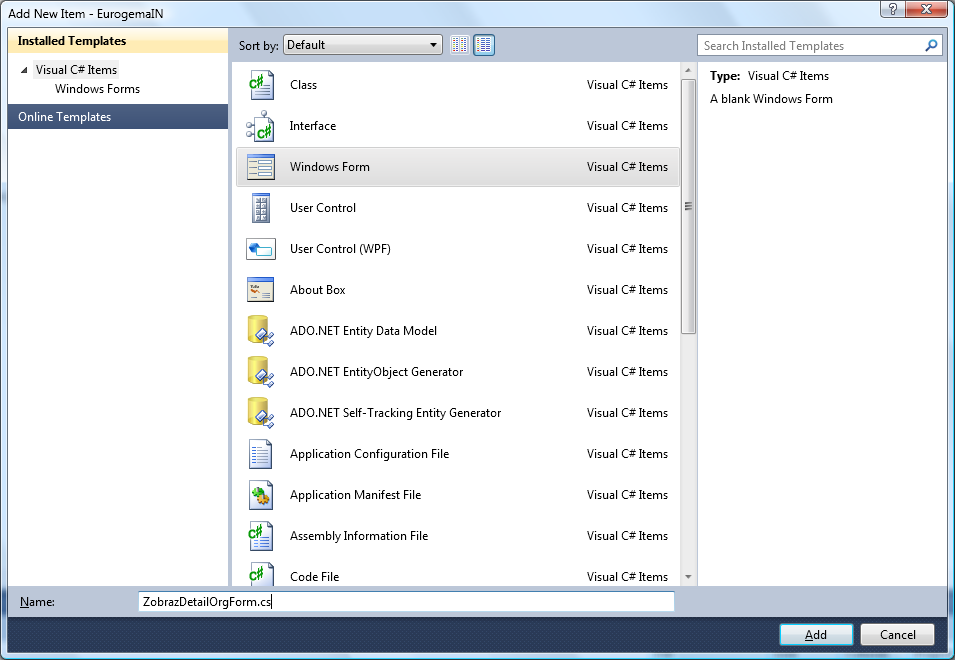
Poté zavoláme funkci CurrentRecordID() rozhraní IHelios, která nám vrátí ID záznamu, nad kterým byla akce vyvolána. Výsledek přiřadíme do proměnné ID.

#### Tvorba formuláře pro zobrazení

Formulář vytvoříme jednoduše tak, že vybereme akci „Windows Form…“ z podnabídky „Add“ z nabídky, která se nám objeví po kliknutí pravým tlačítkem myši na projekt.

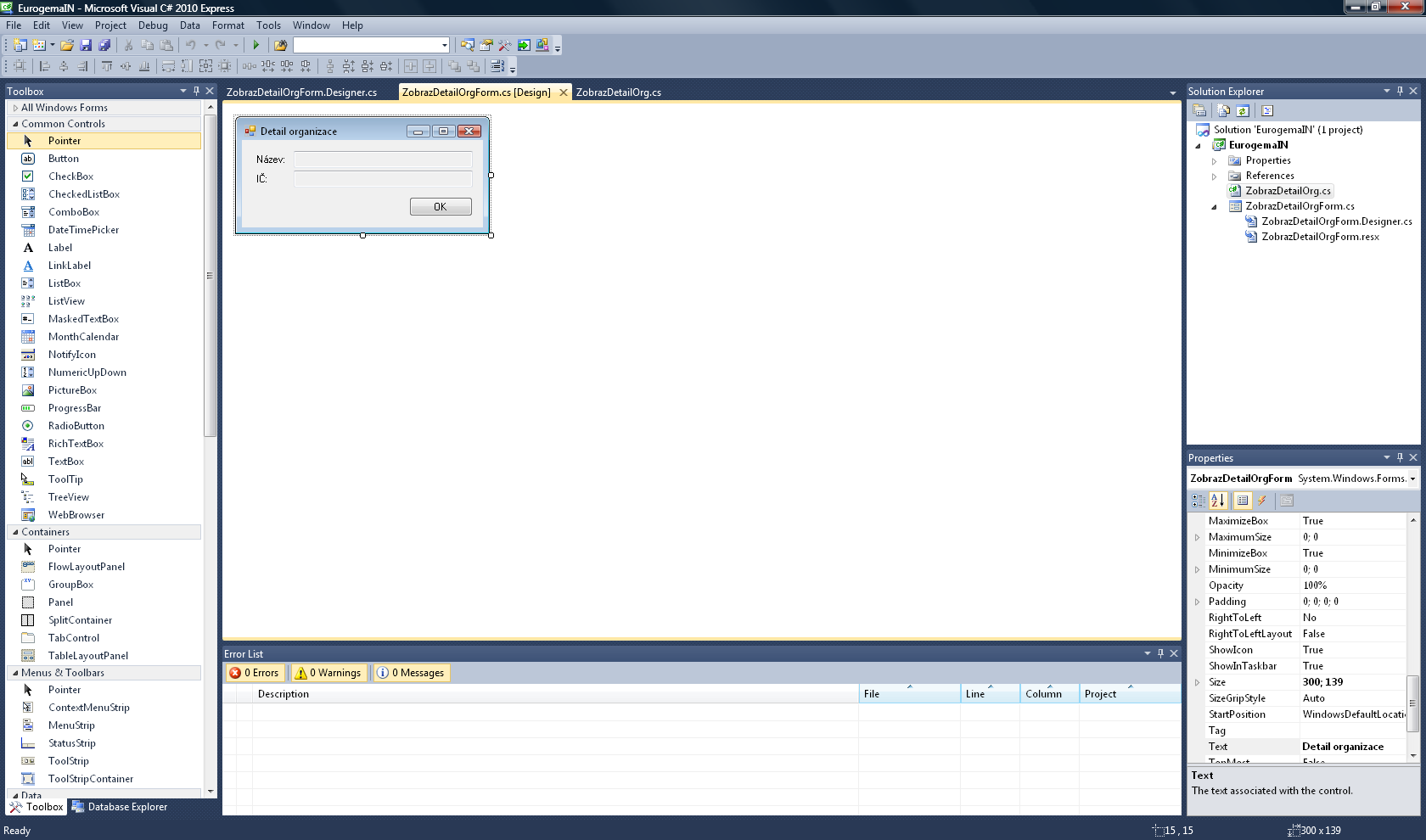


V následné nabídce pojmenujeme formulář ZobrazDetailOrgForm.cs a stiskneme „Add“.



Ve vytvořeném formuláři provedeme následné změny:

* změníme nadpis formuláře na „Detail Organizace“
* změníme výchozí pozici na CenterParent
* přidáme prvky Label „Název:“ a „IČ:“
* přidáme 2x prvek TextBox a nastavíme u nich vlastnost ReadOnly na hodnotu True.
* přidáme tlačítko „OK“ a vytvoříme pro něj Click na zavření okna



V těle formuláře nejprve zapíšeme klauzuli using pro ddPlugin.

Dále si nadefinujeme přetížený konstruktor formuláře se dvěma parametry: Helios typu IHelios a ID typu Int32.

V těle konstruktoru nadefinujeme proměnou Q typu IHeQuery. Tato proměnná nám slouží jako recordset vrácených záznamů z Heliosu.

Vyvoláme proceduru OpenSQL rozhraní IHelios. Tělo této procedury obsahuje klasický SQL příkaz, který vrací záznamy. Návratový typ je IHeQuery. V našem případě bude SQL příkaz vypadat následovně:

"SELECT Firma, ICO FROM TabCisOrg WHERE ID = " + ID.ToString()

Výsledek procedury uložíme do proměnné Q.

Dále přiřadíme pomocí funkce FiledByNameValues hodnoty vrácené SQL dotazem do dříve definovaných textových polí. Funkce se volá nad IHeQuery vráceným z procedury OpenSQL a jako parametr má název sloupce. (V praxi je lepší používat funkci FiledValues, která má za parametr 0-based pořadí sloupců. Je rychlejší). V našem případě je tedy přiřazení následné:

textBox1.Text = Q.FieldByNameValues("Firma");  
textBox2.Text = Q.FieldByNameValues("ICO");

Celá definice pak vypadá následovně:

using System;

using System.Windows.Forms;

using ddPlugin;

namespace EurogemaIN

{

public partial class ZobrazDetailOrgForm : Form

{

public ZobrazDetailOrgForm(IHelios Helios, Int32 ID)

{

this.InitializeComponent();

IHeQuery Q = Helios.OpenSQL("SELECT Firma, ICO FROM TabCisOrg WHERE ID = " + ID.ToString());

textBox1.Text = Q.FieldByNameValues("Firma");

textBox2.Text = Q.FieldByNameValues("ICO");

}

private void button1\_Click(object sender, EventArgs e)

{

Close();

}

}

}

V případě, že bychom chtěli vrátit některé proměnné do hlavního těla programu, můžeme tak učinit standardními způsoby (předání pomocí veřejných proměnných).

#### Zobrazení informací ve formuláři

Samotné vyvolání formuláře provedeme obvyklým způsobem s tím, že do konstruktoru předáme proměnnou s ID aktuálně vybraného řádku a dále pak odkaz na rozhraní IHelios. Námi vytvořené okno nazveme třeba Detail.

Tělo procedury Run pak vypadá následovně:

public void Run(IHelios Helios)

{

Int32 ID = Helios.CurrentRecordID();

ZobrazDetailOrgForm Detaily = new ZobrazDetailOrgForm(Helios, ID);

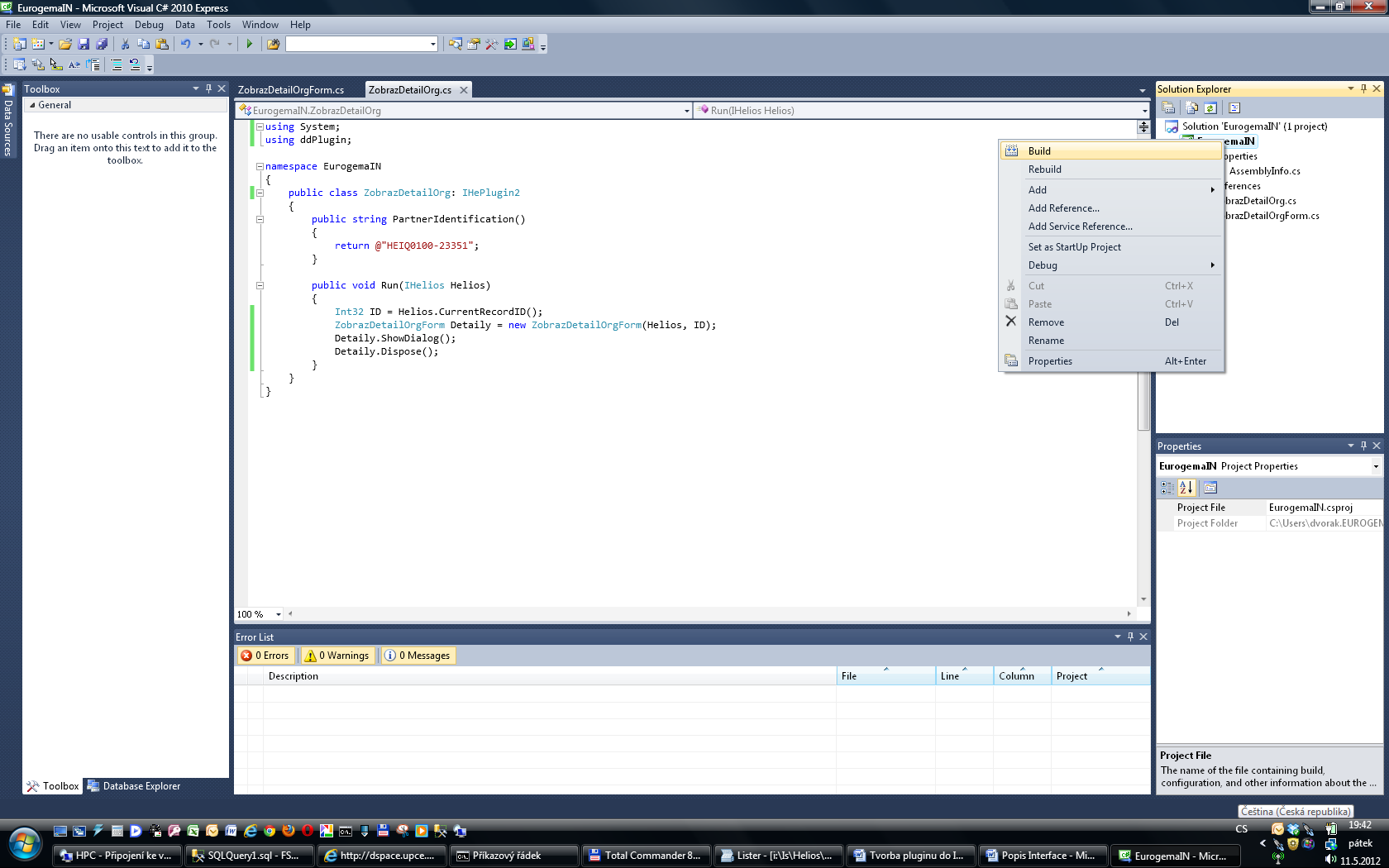
Detaily.ShowDialog();

Detaily.Dispose();

}

## Kompilace knihovny s pluginem

Knihovnu již zkompilujeme klasickým způsobem výběrem akce Build z nabídky po stisknutí pravého tlačítka nad projektem.



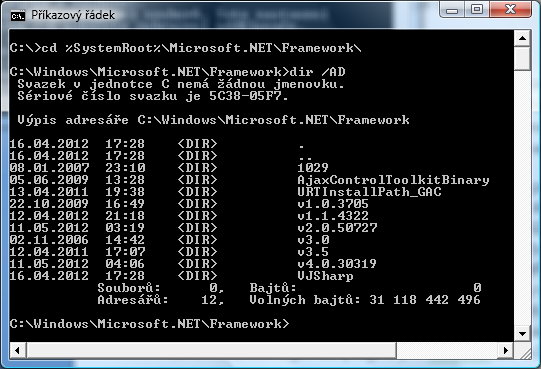
Výsledná knihovna je pak uložena v adresáři s projektem. Pokud jste nic neměnili, pak je to v adresáři "%UserProfile%\Documents\Visual Studio 2010\Projects\EurogemaIN\EurogemaIN\bin\Release".

[Celá projekt je uložen v adresáři Vzor.]

# Registrace knihovny s pluginem

Nejdříve je nutné knihovnu s pluginem překopírovat do adresáře s instalací systému Helios. Je to adresář, ve kterém je soubor Helios.exe, kterým se spouští celý IS. Typicky je to adresář "%ProgramFiles%\LCS International\Helios IQ\" (v případě 64 bitového OS pak "%ProgramFiles(x86)%\LCS International\Helios IQ\").

Poté je nutno na rozdíl od native COM pluginů provést ruční registraci. Tak jak se native com pluginy registrují pomocí Regsvr32.exe, slouží k registraci .NET COM pluginů program RegAsm.exe. Na rozdíl od RegSvr32.exe je však nutné vybrat správnou verzi .NET. Tuto verzi jsme již předvybrali v bodu 3.2 pomocí výběr Target framework (v našem případě tedy .NET 4). V adresáři %SystemRoot%\Microsoft.NET\Framework\ jsou umístěny všechny nainstalované verze .NET. Na nás je pak zvolit tu správnou. Tohle je oproti native COM pluginem velké podstatné omezení, ale nikoli nepřekonatelné. Navíc IS Helios by měl obsahovat automatickou registraci pluginů pro verzi .NET 2, 3 a 3.5.

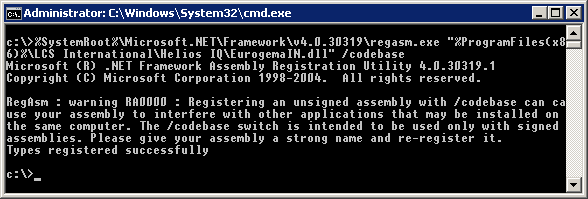


Pro naše účely (.NET 4) zvolíme tedy adresář v4.0.30319.

Registrace pluginu poté spočívá ve spuštění následujícího příkazu:

%SystemRoot%\Microsoft.NET\Framework\v4.0.30319\regasm.exe "%ProgramFiles(x86)%\LCS International\Helios IQ\EurogemaIN.dll" /codebase

Příkaz je opět na jedné řádce a musí se spouštět se zvýšeným oprávněním.



Ještě je třeba podotknout, že ve výchozím nastavení je zakázáno registrovat programy z UNC cest. Proto, aby to bylo umožněno, je třeba přidat do souboru RegAsm.exe.config (soubor se nachází ve stejném adresáři jako utilitka RegAsm.exe) následující řádky:

<runtime>

<loadFromRemoteSources enabled="true"/>

</runtime>

Celý postup registrace může vypadat složitý, ale ve své podstatě je to jeden krátký BAT soubor. Pokud si vytvoříme i soubor regasm.exe.config, který umístíme do stejného adresáře jako knihovnu s pluginem, pak může vypadat následovně (pouze interní příklad):

@Echo off

Echo Registruji interne vytvorene pluginy ...

copy \\HPC\Helios\regasm.exe.config %SystemRoot%\Microsoft.NET\Framework\v4.0.30319\

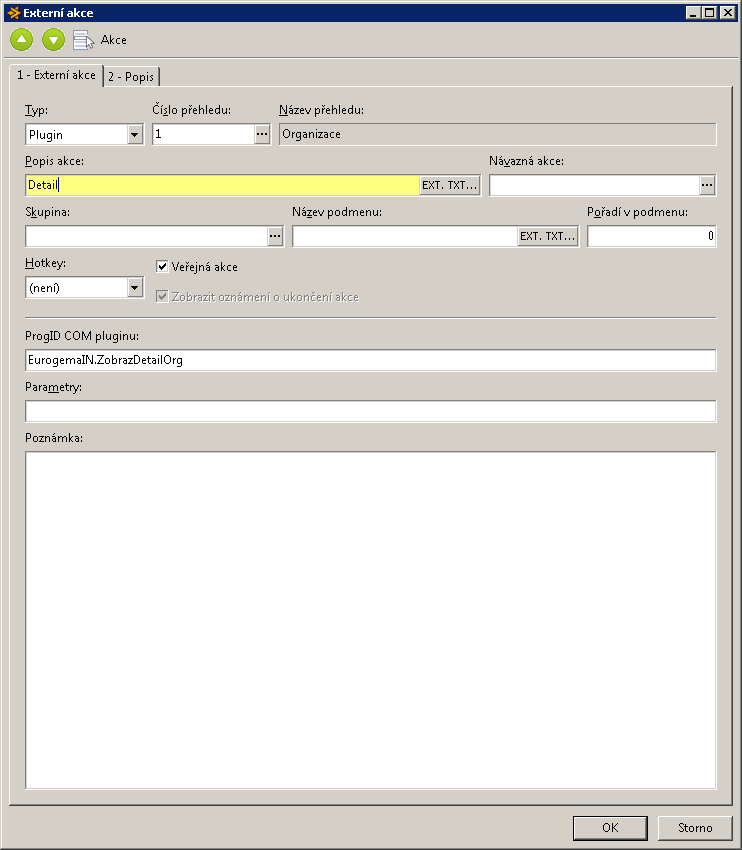
%SystemRoot%\Microsoft.NET\Framework\v4.0.30319\regasm.exe \\HPC\Helios\EurogemaIN.dll /codebase /silent

Echo Hotovo

Pause

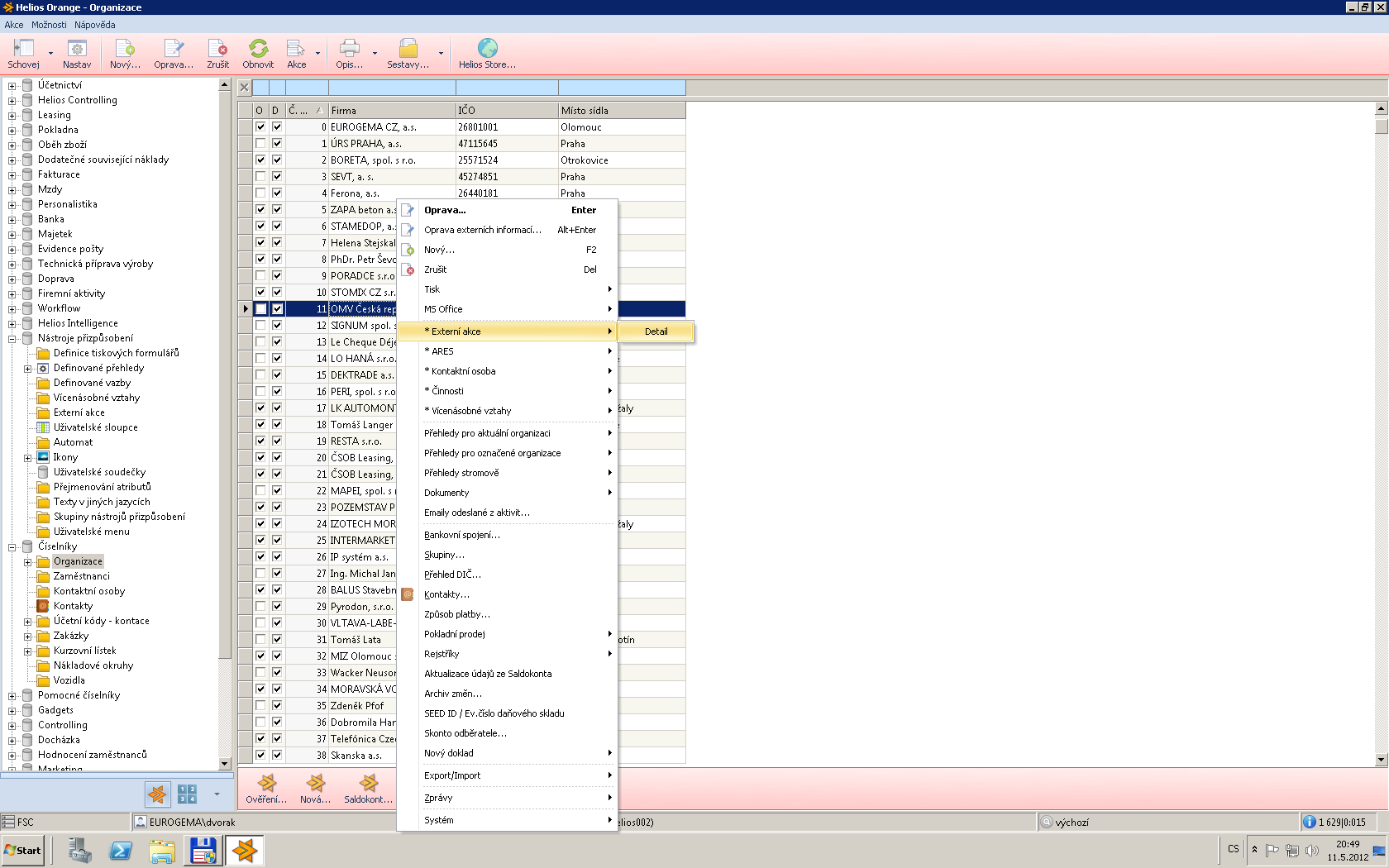
# Použití pluginu v IS Helios

Funkcionalitu pluginu převedeme do systému nadefinováním externí akce.



Důležitou položkou je zde ProgramID COM pluginu. Je to právě jméno třídy, kterou jsme vytvořili v MS Visual Studio, oddělené tečkou od jmeného prostoru, který je zároveň i jménem pluginu.

Po této definici pak již zbývá plugin pouze vyvolat nad sestavou Organizece přes pravé tlačítko myši a zvolením akce Detail z podmenu \*Externí akce.



Pokud jsme vše udělali správně, měl by se objevit následující dialog:

