.NET Framewrok リフレクションの使い方メモ

2012/03/23
method2について、会社でもう少し楽が出来る方法を若様に教わったので修正。

2012/03/21
C# + .NET FrameworkでReflectionを使う際のメモ書き。 うまくまとまった解説をなかなか見つけられなかったので、個人的に調べてわかった範囲でサンプルコードを書いてメモとする。

あるメーカーが提供する製品のSDK□新バージョン毎にSDK用DLLを出してくるので、これを利用するプログラムを毎回リコンパイルする必要があった。 毎回行うのがしんどくなってきたので、リフレクションで解決できないか、というのが動機。

コード

Assembly クラスでターゲットのアセンブリ(DLL)をロードしてアクセスする。

メソッド、プロパティ、フィールドへはInvoke(),InvokeMember()でアクセスする。ただこの際の引数指定が色々面倒。
あとから機能追加するとか、プラグイン形式で機能追加するとか、そういった理由が無いならリフレクションは使わないほうがいい。遅いよ。 それに□IDEでメソッド名とかの補完が出来ないからターゲットのDLLの内容をきちんと押さえておく必要もある。

reflectioncheck.dll

動的呼び出しするDLL□reflectioncheck.dll の名前で生成しておく。

refrectioncheck.cs

```
using System;

namespace reflectionCheck
{
    public enum sampleEnum { tas, low, kau, wah };

    public class positionClass
    {
        public int ipos;

        public positionClass(sampleEnum pos)
        {
            ipos = (int)pos;
        }
    }

    public class targetClass
    {
```

```
        private String str1;

        public String propStr
        {
            get
            {
                return str1;
            }
            set
            {
                str1 = value;
            }
        }

        public String method1(sampleEnum ff)
        {
            return ff.ToString();
        }

        public String method2(sampleEnum[] ff)
        {
            String ans = "";
            for (int i=0; i< ff.Length; i++)
            {
                switch (ff[i])
                {
                    case sampleEnum.kau: ans += "か;"; break;
                    case sampleEnum.low: ans += "ろ;"; break;
                    case sampleEnum.tas: ans += "た;"; break;
                    case sampleEnum.wah: ans += "わ;"; break;
                }
            }
            return ans;
        }

        public positionClass method3(sampleEnum ff)
        {
            positionClass ans = new positionClass( ff );
            return ans;
        }
    }
}
```

reflectioncheckcall.exe

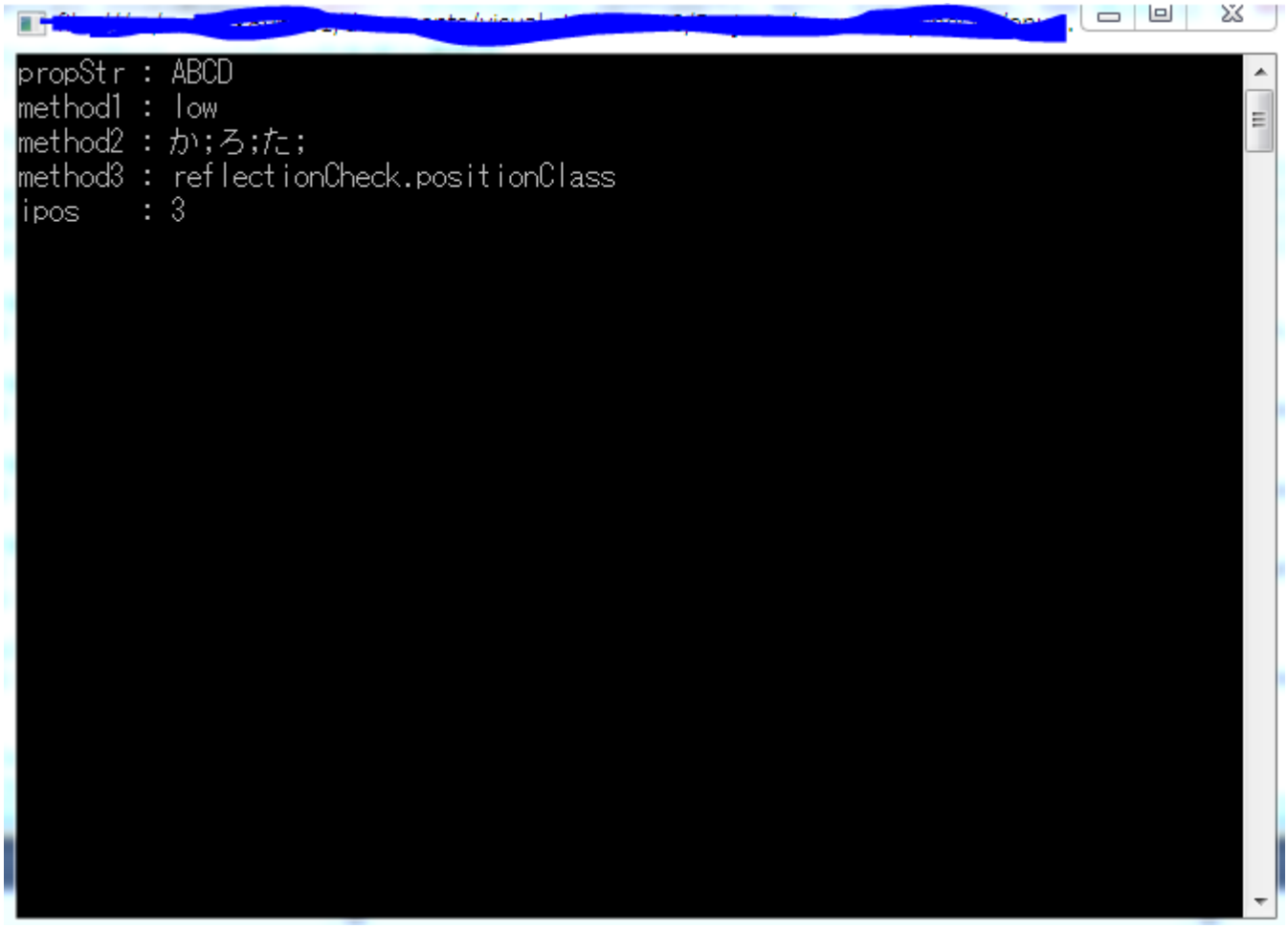
DLLを呼び出すEXE□reflectioncheckcall.exe の名前で生成しておく。

reflectioncall.cs

```
using System;
using System.Collections;
using System.Reflection;
using System.IO;

namespace reflectioncheckcall
{
    class Program
    {
        static void Main(string[] args)
        {
            String here = Path.GetDirectoryName(
Assembly.GetExecutingAssembly().Location ) + @"\reflectioncheck.dll"; //
同じフォルダにDLLがあることを前提とする
            Assembly reflectionCheckDLL = Assembly.LoadFrom(here);

            Type TtargetClass =
reflectionCheckDLL.GetType("reflectionCheck.targetClass",true);
            Type TpositionClass =
reflectionCheckDLL.GetType("reflectionCheck.positionClass", true);
            Type TsampleEnum =
reflectionCheckDLL.GetType("reflectionCheck.sampleEnum", true);

            // reflectionCheck.targetClass を生成(コンストラクタなし)
            Object oTargetClass =
Activator.CreateInstance(TtargetClass, true);

            // reflectionCheck.positionClass を生成(コンストラクタあり)
            Object oPositionClass5 = TpositionClass.InvokeMember(null,
BindingFlags.CreateInstance, null, null, new Object[] {
TsampleEnum.InvokeMember("wah", BindingFlags.GetField, null, null,
null) });

            //生成したtargetClassのpropStrプロパティに設定
            TtargetClass.InvokeMember("propStr",
BindingFlags.SetProperty, null, oTargetClass, new Object[] { "ABCD" });

            // reflectionCheck.sampleEnumの生成と値設定
            Object oSampleEnum = TsampleEnum.InvokeMember("low",
BindingFlags.GetField, null, null, null);

            // reflectionCheck.sampleEnum[]の生成と値設定
            ArrayList oSampleEnums = new ArrayList();
            oSampleEnums.Add(TsampleEnum.InvokeMember("kau",
BindingFlags.GetField, null, null, null));
            oSampleEnums.Add(TsampleEnum.InvokeMember("low",
BindingFlags.GetField, null, null, null));
            oSampleEnums.Add(TsampleEnum.InvokeMember("tas",
BindingFlags.GetField, null, null, null));

            // targetClassのpropStrプロパティ読み出し
```

```
            String s1 = (String)TtargetClass.InvokeMember("propStr",
BindingFlags.GetProperty, null, oTargetClass, null);

            // targetClassのmethod1メソッド呼び出し
            String s2 = (String)TtargetClass.InvokeMember("method1",
BindingFlags.InvokeMethod, null, oTargetClass, new Object[] {
oSampleEnum });

            // targetClassのmethod2メソッド呼び出し
            String s3 = (String)TtargetClass.InvokeMember("method2",
BindingFlags.InvokeMethod, null, oTargetClass, new Object[] {
oSampleEnums.ToArray(TsampleEnum) });

            // targetClassのmethod3メソッド呼び出し
            Object oPositionClass4 =
TtargetClass.InvokeMember("method3", BindingFlags.InvokeMethod, null,
oTargetClass, new Object[] { oSampleEnum });
            String s4 = oPositionClass4.ToString();

            // positionClassのiposフィールド読み出し
            String s5 = ((int)TpositionClass.InvokeMember("ipos",
BindingFlags.GetField, null, oPositionClass5, null)).ToString();


            Console.WriteLine("propStr : {0}", s1);
            Console.WriteLine("method1 : {0}", s2);
            Console.WriteLine("method2 : {0}", s3);
            Console.WriteLine("method3 : {0}", s4);
            Console.WriteLine("ipos    : {0}", s5);

            return;
        }
    }
}
```

実行結果



技術資料, windows, .NET