



Full Life-Cycle Development & Implementation for Business Intelligence, Data Warehousing, and Corporate Performance Management.

# 2007

## Designer SDK with C# UniverseRetrieveInfo()

# Actionable Intelligence

Web intelligent Design, Inc.

[www.webidesign.com](http://www.webidesign.com)

Rosalind Beasley, PMP

[Rosalind@webidesign.com](mailto:Rosalind@webidesign.com)

5/1/2007

The first step is to create a reference in your project to the Business Objects Designer Object Library. This is done by right mouse clicking on the References folder in the Solution Explorer and choosing Add Reference. This brings up the Tab Dialog below. Choose the COM Tab and pick Business Objects Designer 6.5 Objects Library.

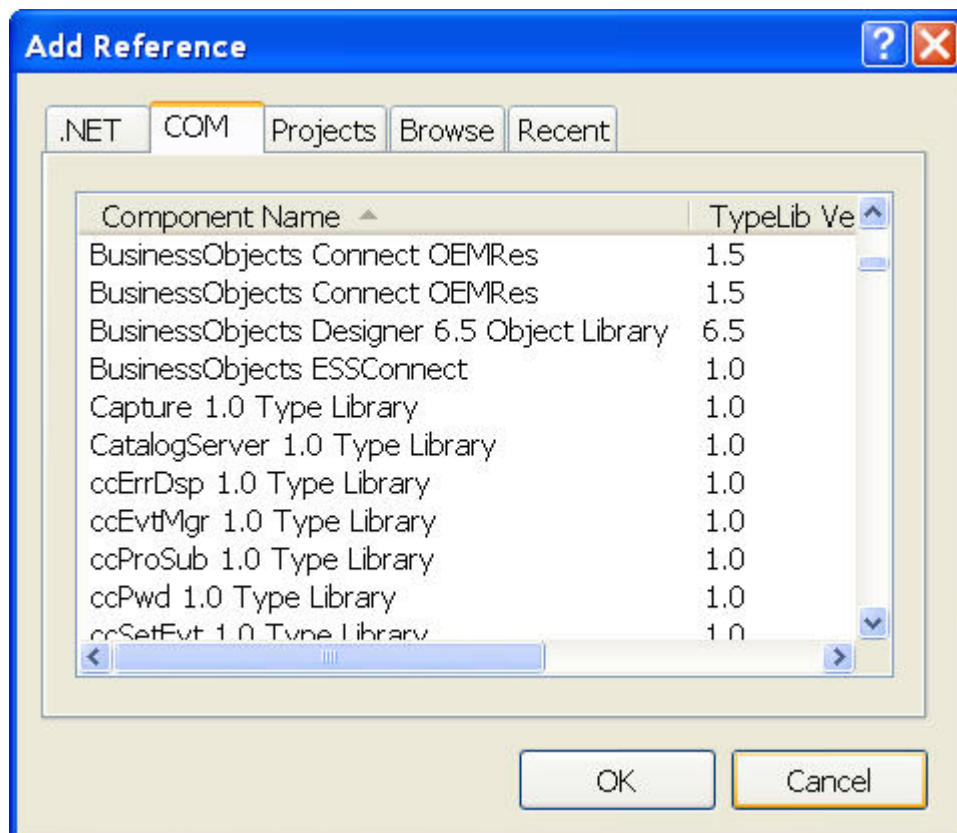


Figure 1 adding a BO Designer Reference

This action puts an Interop.Designer.dll into your bin directory so you can manipulate Designer.

Next we add a new class. This is done by right mouse clicking in the Solution Explorer, choosing Add Item and then choosing Class. Give the new class a name, i.e., clsBOUnvManager.

The following code sample describes the implementation of the UniverseRetrieveInfo method within the clsBOUnvManager class. Read it carefully and you will have a good understanding of how to retrieve information from a Business Objects universe from a C# application. This code assumes that you have already logged into Designer and opened a universe. See previous postings for examples of designerLogin and universeOpen methods.

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Collections;
namespace webdesign
{
    public class clsBOUnvManager
    {
        public static Designer.Application dsApp;
        public static Designer.Universe dsUnv;
        public static Hashtable colCls;
        public static Designer.Class dscls ;
        public static Designer.Classes dsclss;
        public static Designer.Universes dsUnvs;

        private static bool UniverseRetrieveInfo(Designer.Universe dsUnv)
        {
            //create a hashtable to hold the collection of universe objects
            clsBOUnvManager.colCls = new Hashtable();
            //loop through the universe classes
            for (int i = 1; i <= clsBOUnvManager.dsUnv.Classes.Count; i++)
            {
                //add each object within the current class to the hashtable
                clsBOUnvManager.colCls.Add(i, clsBOUnvManager.dsUnv.Classes.get_Item(i));
                //the universe may contain sub-class so we need to recurse the structure
                bool RecurseComplete =clsBOUnvManager.ClassRecurse(clsBOUnvManager.dsUnv.Classes.get_Item(i), false);
            }
            return true;
        }

        private static Boolean ClassRecurse(Designer.Class boclsRoot, bool RecurseComplete)
        {
            //this method excepts two parameters ( the root class, and a flag which indicates whether the end of the class structure has been reached
            // this method loops through the universe class structure recursively untill it reaches the end of the tree
            Designer.Class clsCurrent;
            if (boclsRoot.Classes.Count != 0)
            {
                for (int i = 1; i <= boclsRoot.Classes.Count; i++)
                {
                    clsCurrent = boclsRoot.Classes.get_Item(i);
                    clsBOUnvManager.colCls.Add(clsCurrent.Name, clsCurrent);
                    Boolean x = ClassRecurse(clsCurrent, false);
                }
            }
            return true;
        }
    }
}
```