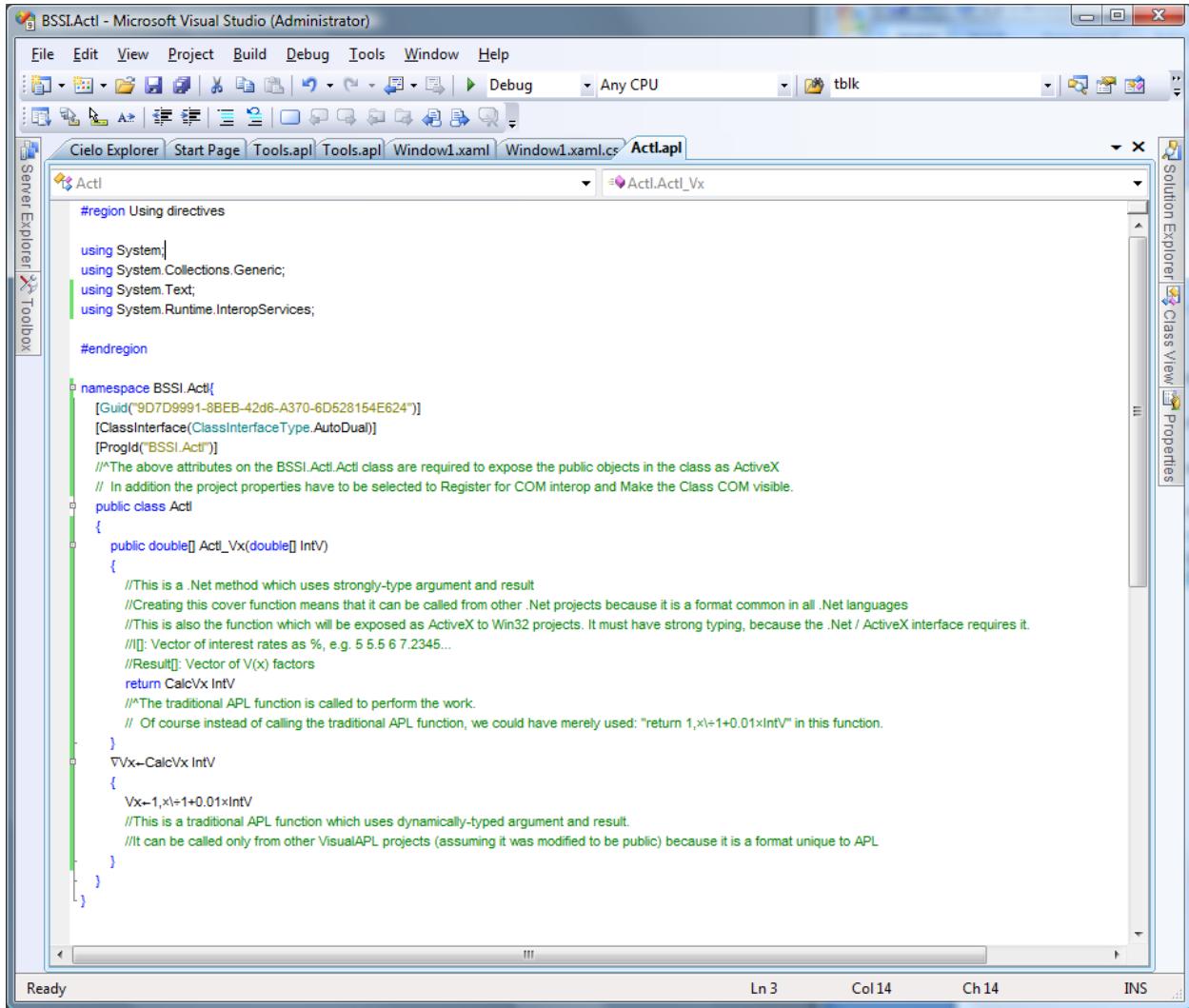


VisualAPL for Visual Studio 2008: Using VisualAPL in APL+Win and in C# WPF GUI

The following screen capture illustrates a simple example of a VisualAPL class 'BSSI.Actl.Actl', which includes the method 'Actl_Vx', which can be referenced and used by any .Net project. The class is also exposed as ActiveX so it can be used by any Win32 application such as APL+Win or VB6.



The screenshot shows the Microsoft Visual Studio 2008 IDE interface. The title bar reads "BSSI.Actl - Microsoft Visual Studio (Administrator)". The menu bar includes File, Edit, View, Project, Build, Debug, Tools, Window, Help. The toolbar has various icons for file operations. The status bar at the bottom shows "Ready", "Ln 3", "Col 14", "Ch 14", and "INS". The code editor window displays the following VisualAPL code:

```
#region Using directives

using System;
using System.Collections.Generic;
using System.Text;
using System.Runtime.InteropServices;

#endregion

namespace BSSI.Actl{
    [Guid("9D7D9991-8BEB-42d6-A370-6D528154E624")]
    [ClassInterface(ClassInterfaceType.AutoDual)]
    [ProgId("BSSI.Actl")]
    //^The above attributes on the BSSI.Actl.Actl class are required to expose the public objects in the class as ActiveX
    // In addition the project properties have to be selected to Register for COM interop and Make the Class COM visible.
    public class Actl
    {
        public double[] Actl_Vx(double[] IntV)
        {
            //This is a .Net method which uses strongly-type argument and result
            //Creating this cover function means that it can be called from other .Net projects because it is a format common in all .Net languages
            //This is also the function which will be exposed as ActiveX to Win32 projects. It must have strong typing, because the .Net / ActiveX interface requires it.
            //[]: Vector of interest rates as %, e.g. 5 5.5 6 7.2345...
            //Result[]: Vector of V(x) factors
            return CalcVx(IntV);
            //^The traditional APL function is called to perform the work.
            // Of course instead of calling the traditional APL function, we could have merely used: "return 1,x\=1+0.01\x\IntV" in this function.
        }
        \Vx←CalcVx IntV
        {
            Vx←1,x\=1+0.01\x\IntV
            //This is a traditional APL function which uses dynamically-typed argument and result.
            //It can be called only from other VisualAPL projects (assuming it was modified to be public) because it is a format unique to APL
        }
    }
}
```

The following screen capture illustrates the use of the VisualAPL 'BSSI.Actl.Actl' class as ActiveX by an APL+Win session.

```

    sys
0 2 0 1 1 1000 1 0 0 0 100 2 1 0 0 0 0 200 6202 37 0 0 -1 0
    #'@wi 'XInfo' 'BSSI.Actl'
BSSI.Actl ActiveObject BSSI.Actl.Actl
    A←'A'@wi 'Create' 'BSSI.Actl'
    A@wi 'methods'
Close Create Defer Delete EnumEnd EnumNext Event Exec Info Modify New
    Open Ref Send Set SetLinks XActl_Vx XEquals XGetHashCode XGetType
    A@wi '??XActl_Vx'
XActl_Vx method:
    Result@Array_Double ← @WI 'XActl_Vx' IntV@Array_Double
        v1←A@wi 'XActl_Vx' (120p5.25)
    v2←1,×\÷120p1.0525
        v2≡v1
1
    v1
1 0.95011876484561 0.90272566731174 0.85769659602066 0.81491363042343
    0.77426473199375 0.73564345082542 0.69894864686501 0.66408422504989
    0.63095888365785 0.59948587520936 0.5695827792963 0.54117128674233
    0.51417699452953 0.48852921095442 0.46416077050301 0.44100785796011
    0.41900984129227 0.39810911286676 0.37825093859075 0.35938331457553
    0.34145683095062 0.3244245424709 0.30824184557805 0.29286636159435
    0.27825782574285 0.26437798170342 0.25119048142843 0.23866078995575
    0.22675609496983 0.21544522087395 0.20469854714865 0.19448793078256
    0.18478663257251 0.17556924709977 0.16681163619931 0.15849086574756
    0.15058514560338 0.14307377254478 0.13593707605205 0.12915636679529
    0.12271388769149 0.11659276740284 0.11077697615472 0.10525128375745
    0.10000121972205 0.095013035365364 0.090273667805572 0.085770705753512
    0.081492357010463 0.077427417587138 0.073565242363077 0.069895717209574
    0.066409232503158 0.063096657960245 0.059949318727074 0.056958972662303
    0.054117788752782 0.051418326605969 0.048853516965291 0.046416643197426
    0.044101323703018 0.041901495204767 0.039811396869138 0.037825555220083
    0.035938769805305 0.034146099577487 0.032442849954857 0.03082456052718
    0.029286993374993 0.02782612197149 0.026438120637995 0.025119354525411
    0.023866370095402 0.022675886076392 0.021544784870681 0.020470104390196
    0.019449030299473 0.01847888864558 0.017557138855658 0.016681367083761
    0.015849279889559 0.01505869823236 0.014307551764713 0.013593873410653
    0.012915794214397 0.012271538445983 0.011659418951053 0.011077832732592
    0.010525256753056 0.010000243945897 0.0095014194260303 0.00902747688934
    0.0085771751917719 0.0081493350990707 0.007742836198642 0.0073566139654556
    0.0069896569743046 0.0066410042511208 0.0063097427564093
    0.0059950049942131 0.005695966740345 0.0054118448839382 0.0051418953766634
    0.0048854112842408 0.0046417209351456 0.0044101861616585
    0.0041902006286542 0.0039811882457522 0.0037826016586719
    0.0035939208158403 0.0034146516064991 0.003244324566745 0.003082493650114
    0.0029287350594908 0.0027826461372834 0.0026438443109581
    0.0025119660911716 0.002386666119878 0.0022676162659173 0.0021545047657172

```

The Visual Studio 2008 solution containing the VisualAPL ‘BSSI.Actl.Actl’ class also contains two other projects, a C# Windows Presentation Foundation (WPF) GUI project ‘TestHarness’ which uses the VisualAPL ‘BSSI.Actl.Actl’ class to perform the calculations and the VisualAPL ‘BSSI.APLTools.Tools’ class to facilitate the validation of the GUI entry fields. The following screen capture illustrates the WPF GUI and the VIsualAPL calculated result.

The screenshot shows a WPF application window titled "Window1". The main title bar has standard minimize, maximize, and close buttons. The window content is titled "Interest Only Present Value". It contains four input fields with their respective values:

Input Field	Value
Interest Rate as % (e.g. 5.25)	5.25
Current Age (Integer)	45
Retirement Age (Integer)	65
Fund Present Value (\$Amt)	250000

Below these fields is a button labeled "Calculate Projected Accumulated Value". To the right of the button, the calculated result "695636.08" is displayed. The entire application interface is contained within a single window frame.