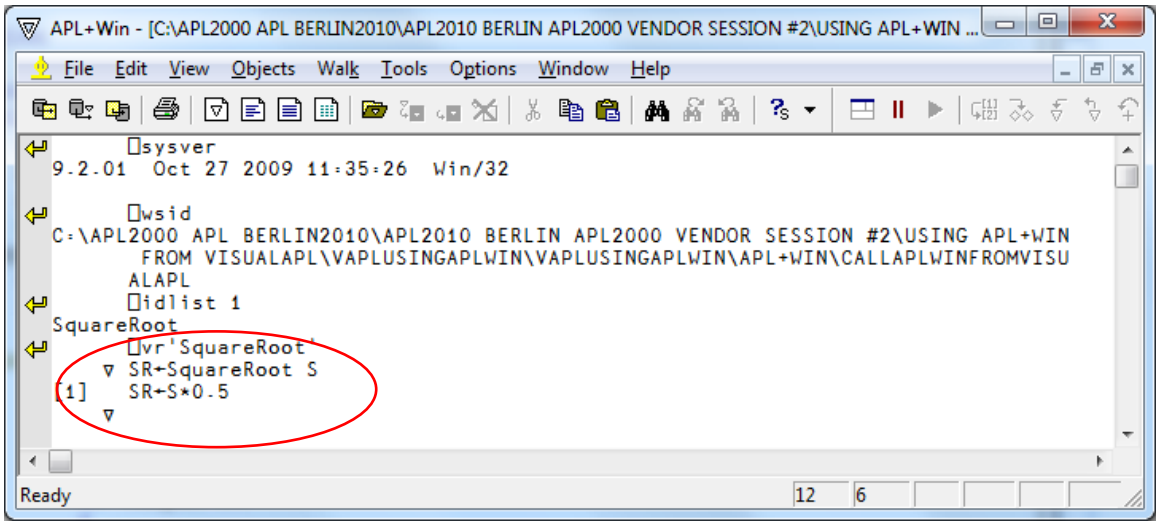
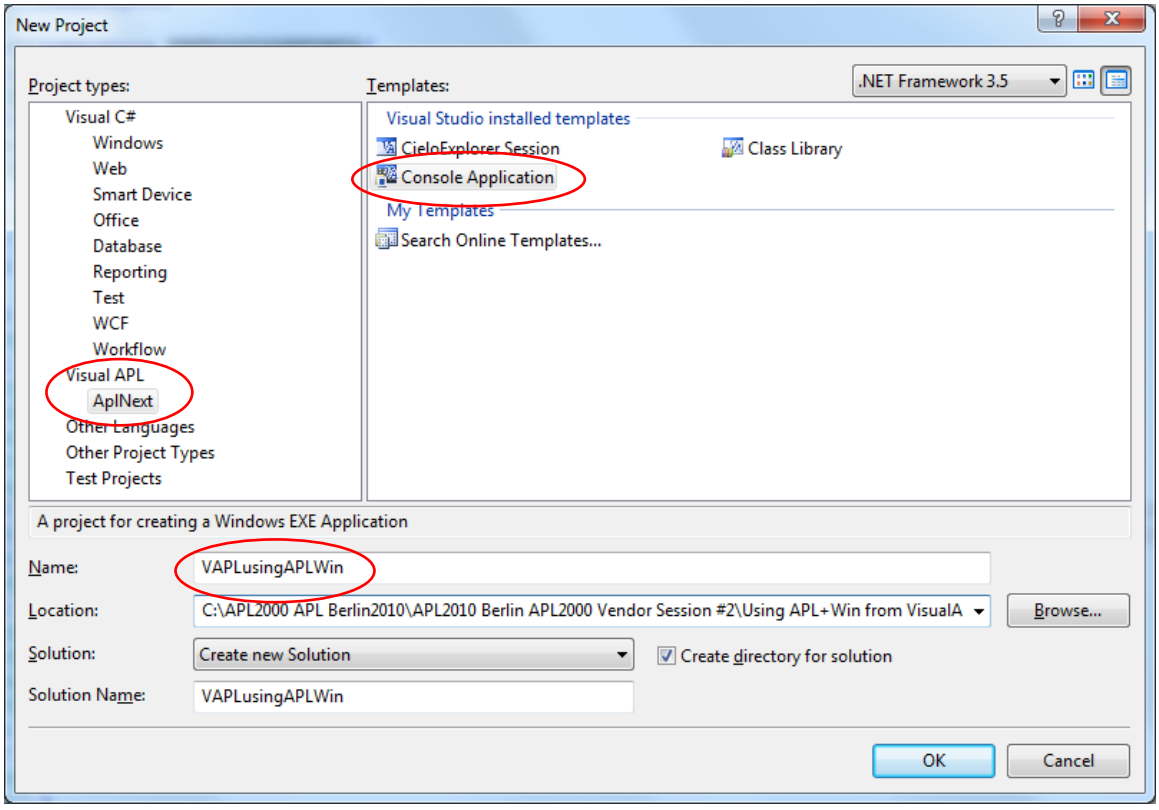
Using APL+Win from VisualAPL

APL+Win is exposed as an ActiveX server when APL+Win is registered on the workstation. Any .Net programming language, e.g. C#, VB.Net and VisualAPL, can access ActiveX objects such as APL+Win.

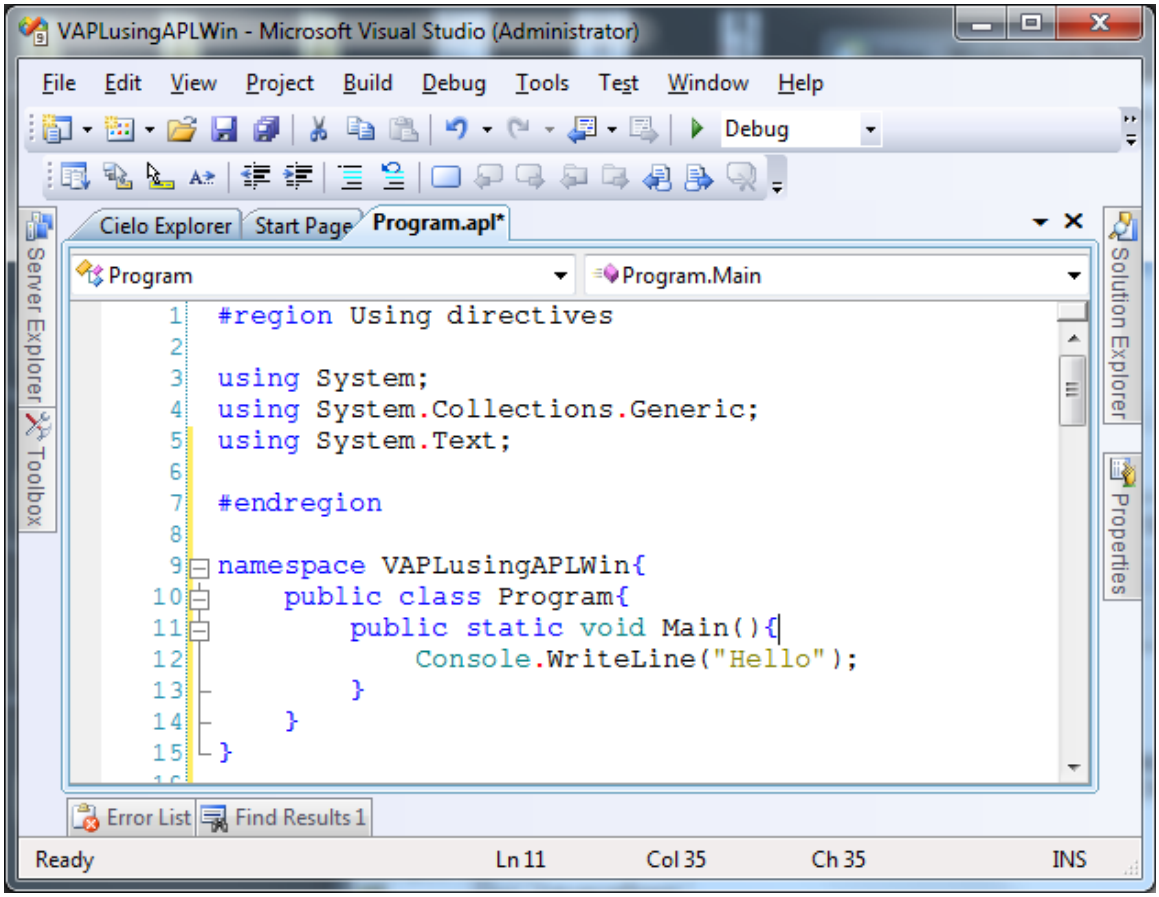
First create an application-specific APL+Win workspace and an associated APL+Win function which will be called from VisualAPL. Make sure that the workspace name is “CALLAPLWINFROMVISUALAPL”.



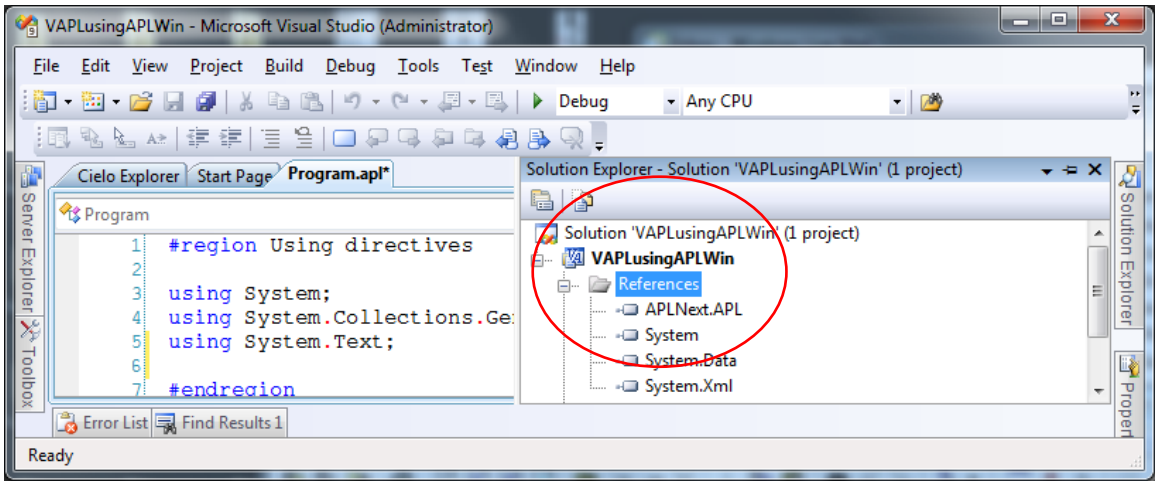
Next create the VisualAPL project which will call APL+Win as an ActiveX server. Open an instance of Microsoft Visual Studio 2008 and use the File > New > Project > VisualAPL > APLNext > Console Application to create the a simple demonstration project illustrating the concept.



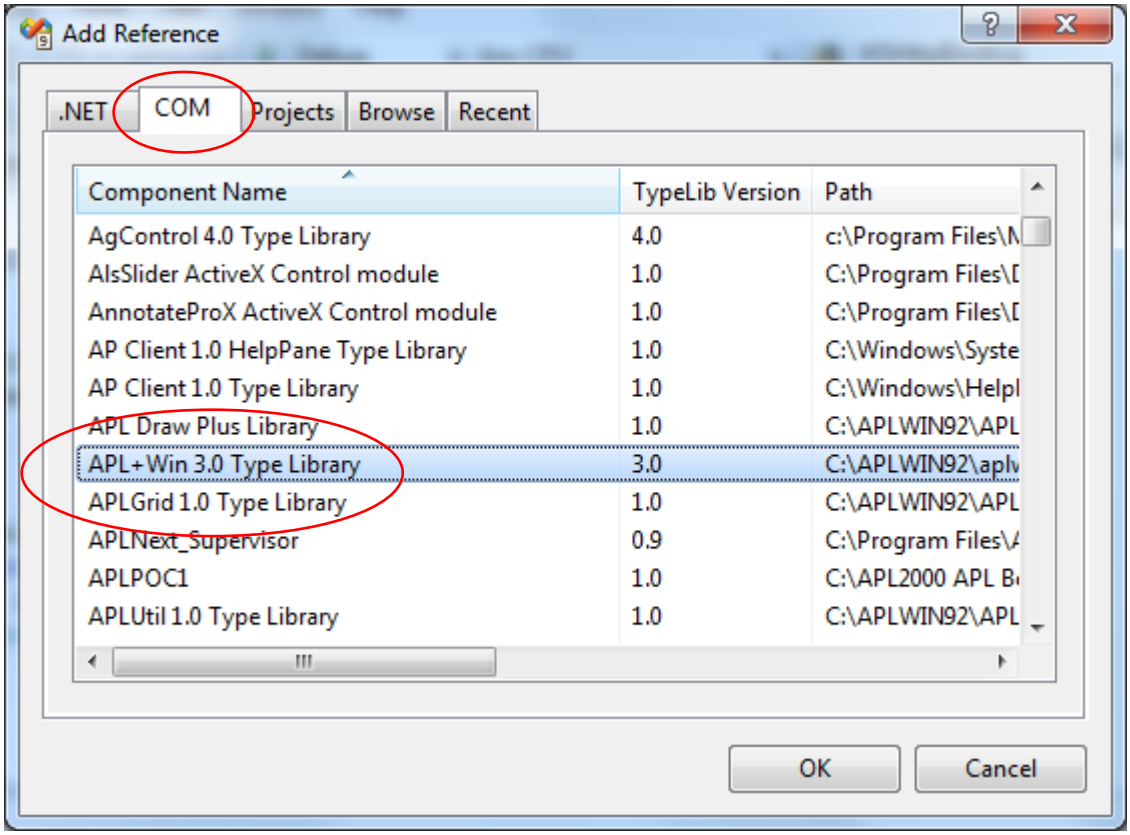
The VisualAPL Console Application template will create the following Program.apl source code file:



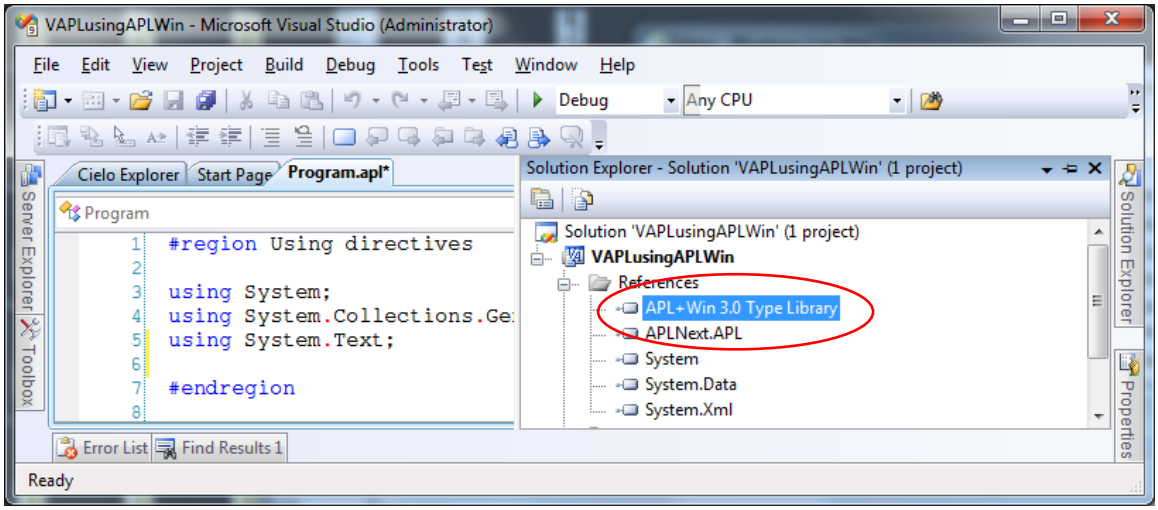
Use the Solution Explorer tab to open the References node:



Right click the References node and select Add A Reference to present the Add Reference dialog. Click the COM tab and select the APL+Win 3.0 Type Library aplwco.dll file.



The APL+Win 3.0 Type Library reference will now exist in the Solution Explorer. This means that a copy of the applicable aplwco.dll will be included in this VisualAPL project and the methods, properties and events of this library will be available for use in the VisualAPL project.

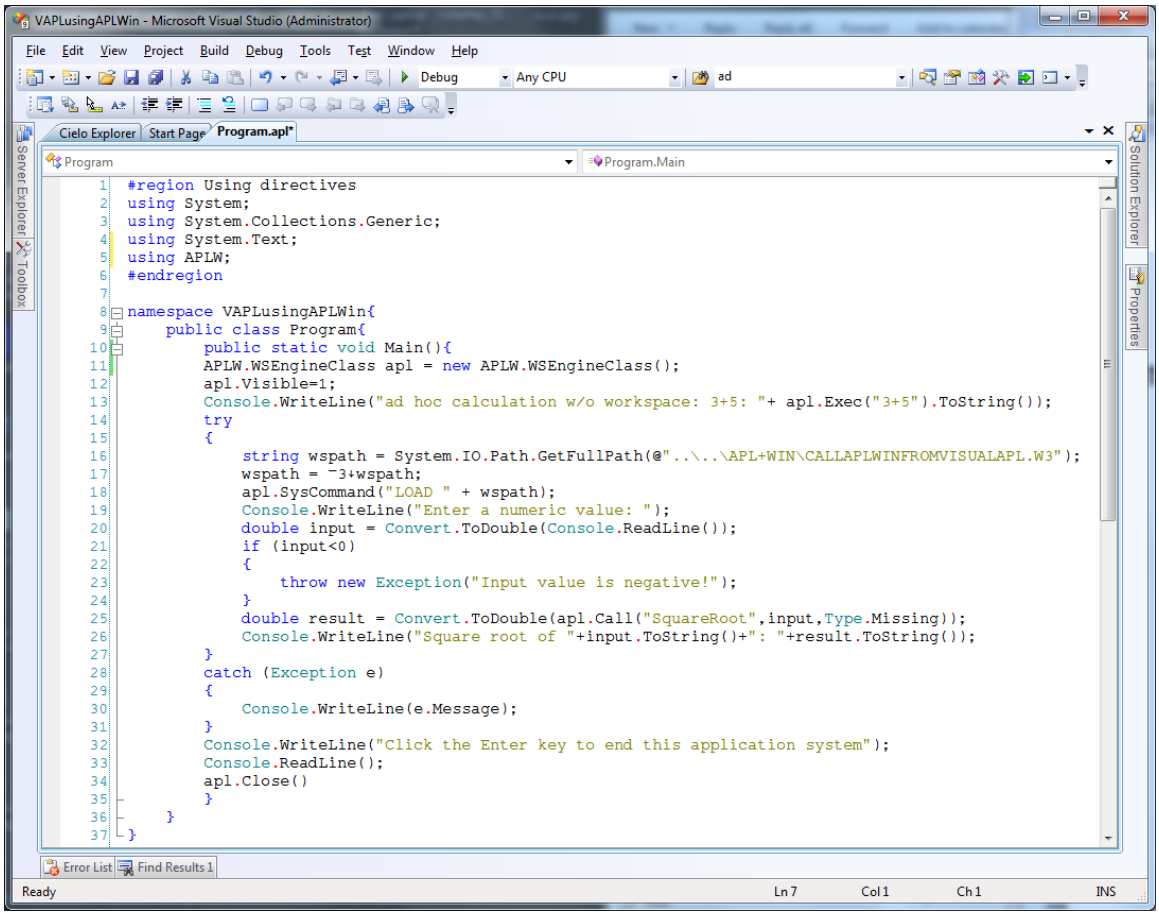


Copy the following APL+Win source code replacing the default code in the VisualAPL project file Program.apl.

```
#region Using directives
using System;
using System.Collections.Generic;
using System.Text;
using APLW;
#endregion

namespace VAPLusingAPLWin{
      public class Program{
            public static void Main(){
            APLW.WSEngineClass apl = new APLW.WSEngineClass();
            apl.Visible=1;
            Console.WriteLine("ad hoc calculation w/o workspace: 3+5:
"+ apl.Exec("3+5").ToString());
            try
        {
            string wspath =
System.IO.Path.GetFullPath(@"..\..\APL+WIN\CALLAPLWINFROMVISUALAPL.W3
");
            wspath = ¯3↓wspath;
            apl.SysCommand("LOAD " + wspath);
            Console.WriteLine("Enter a numeric value: ");
            double input = Convert.ToDouble(Console.ReadLine());
            if (input<0)
            {
                throw new Exception("Input value is negative!");
            }
            double result =
Convert.ToDouble(apl.Call("SquareRoot",input,Type.Missing));
            Console.WriteLine("Square root of "+input.ToString()+":
"+result.ToString());
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
            Console.WriteLine("Click the Enter key to end this
application system");
            Console.ReadLine();
            apl.Close()
            }
      }
}
```

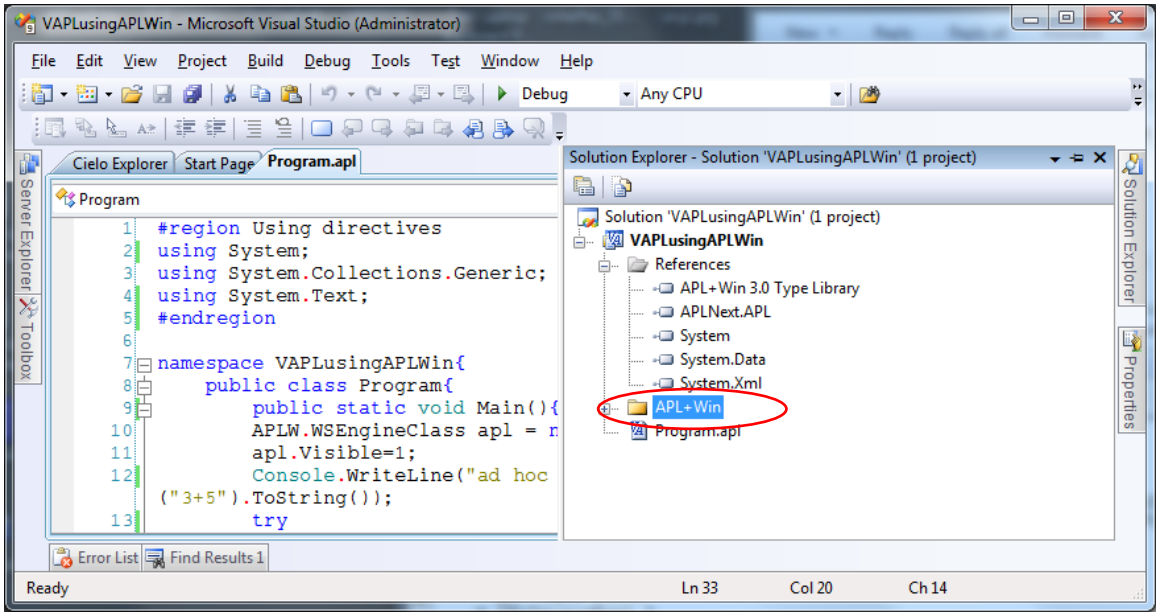
The VisualAPL Program.apl source should now look like:



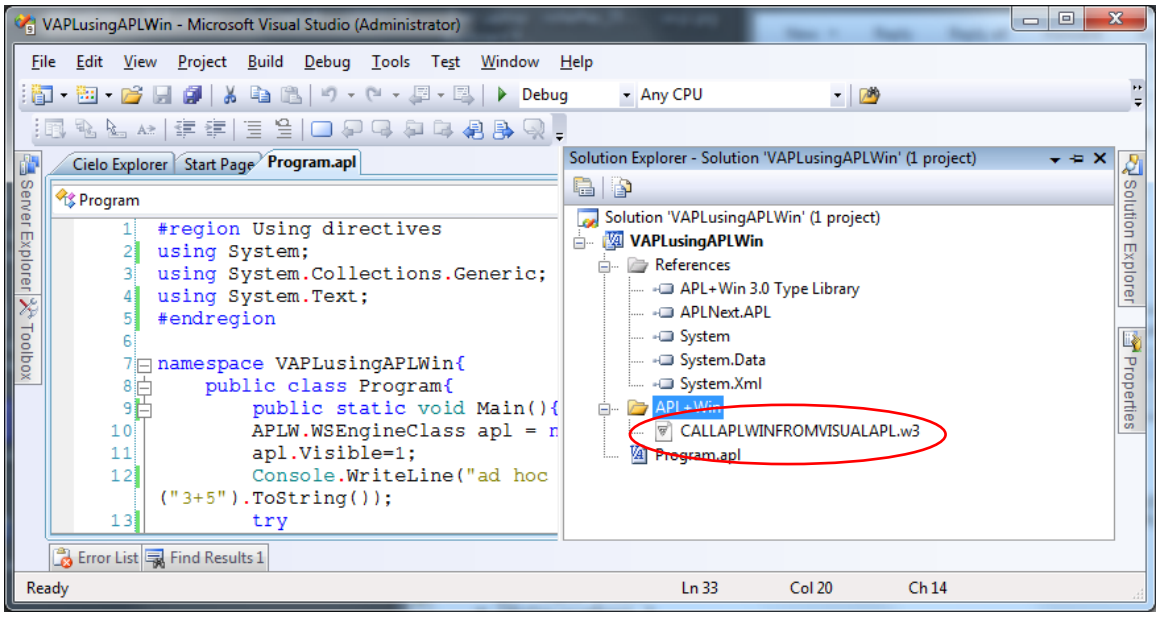

```
#region Using directives
using System;
using System.Collections.Generic;
using System.Text;
using APLW;
#endregion

namespace VAPLusingAPLWin{
    public class Program{
        public static void Main(){
        APLW.WSEngineClass apl = new APLW.WSEngineClass();
        apl.Visible=1;
        Console.WriteLine("ad hoc calculation w/o workspace: 3+5: "+ apl.Exec("3+5").ToString());
        try
        {
            string wspath = System.IO.Path.GetFullPath(@"..\..\APL+WIN\CALLAPLWINFROMVISUALAPL.W3");
            wspath = ¯3↓wspath;
            apl.SysCommand("LOAD " + wspath);
            Console.WriteLine("Enter a numeric value: ");
            double input = Convert.ToDouble(Console.ReadLine());
            if (input<0)
            {
                throw new Exception("Input value is negative!");
            }
            double result = Convert.ToDouble(apl.Call("SquareRoot",input,Type.Missing));
            Console.WriteLine("Square root of "+input.ToString()+": "+result.ToString());
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
        Console.WriteLine("Click the Enter key to end this application system");
        Console.ReadLine();
        apl.Close()
        }
    }
}
```

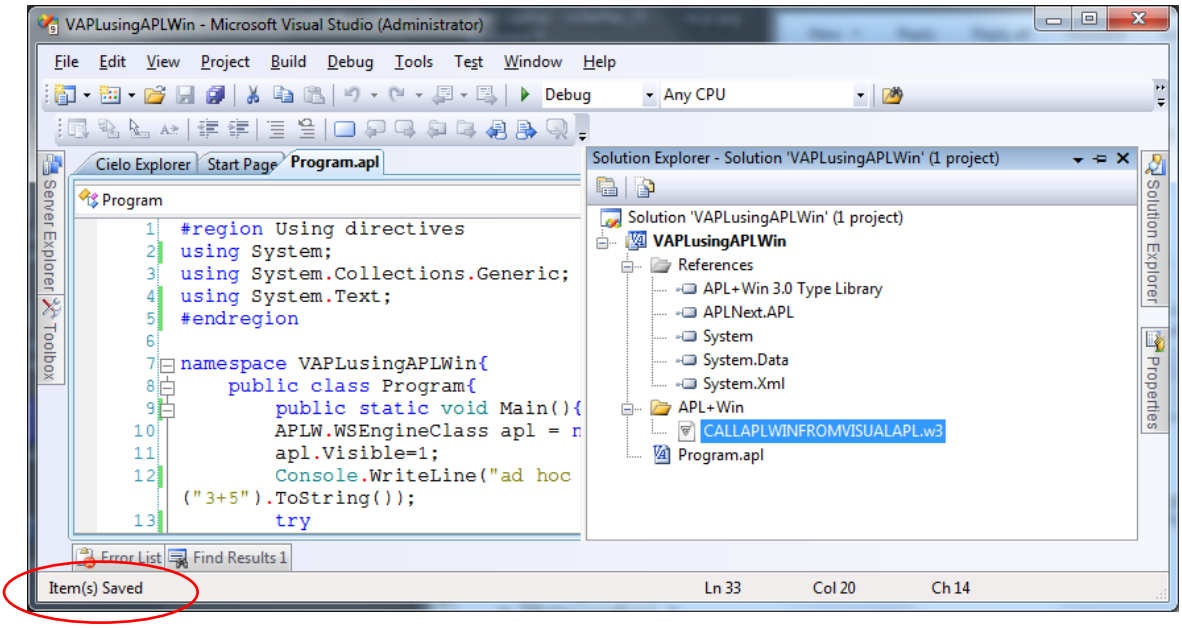
In the Visual Studio Solution Explorer, right click the VisualAPL project name and select Add > New Folder and name it “APL+Win”.



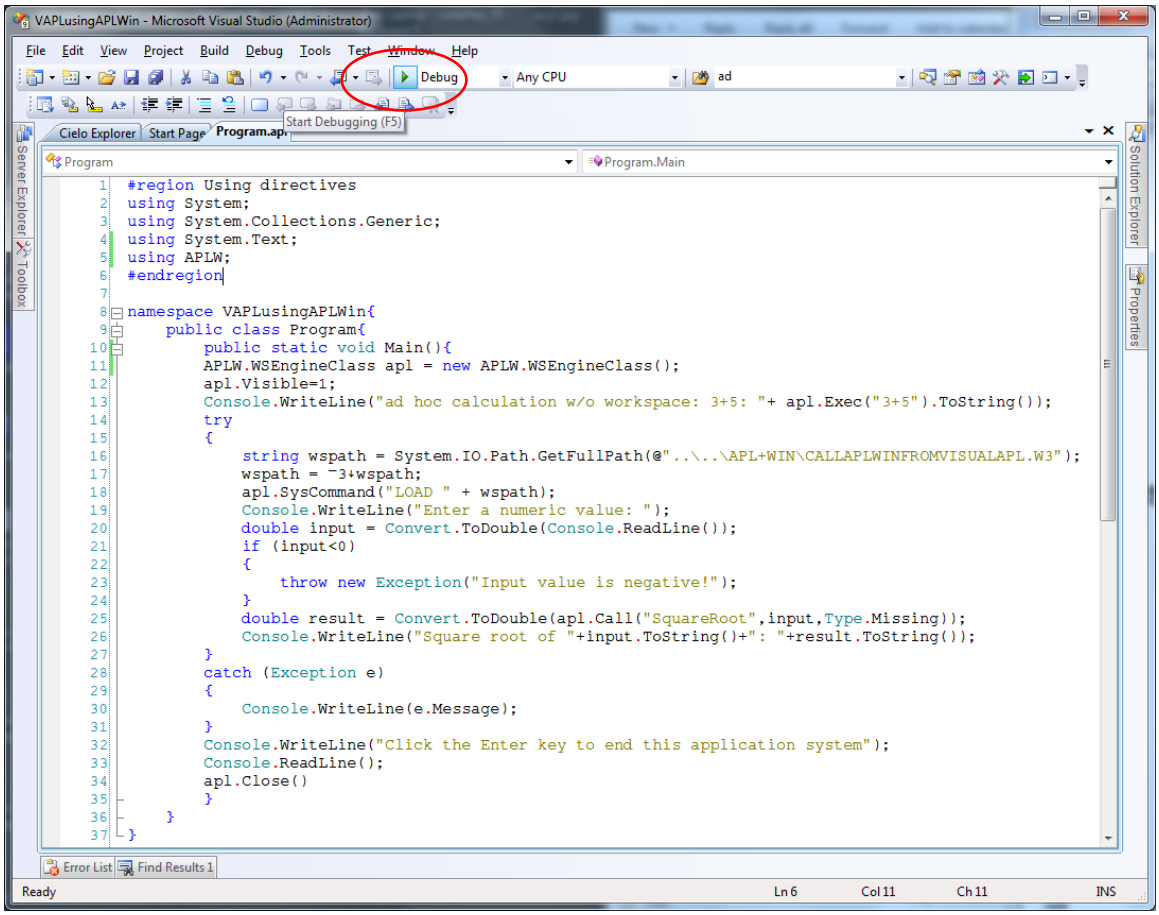
In the Visual Studio Solution Explorer right click the APL+Win folder and select Add > Existing Item. Browse to the location of the APL+Win workspace created for this project and add a copy of this workspace file to the APL+Win folder in the VisualAPL project.



Use the Visual Studio File > Save All option to save all files in this solution. The saved status of the solution files will be indicated in the status bar of the Visual Studio window.



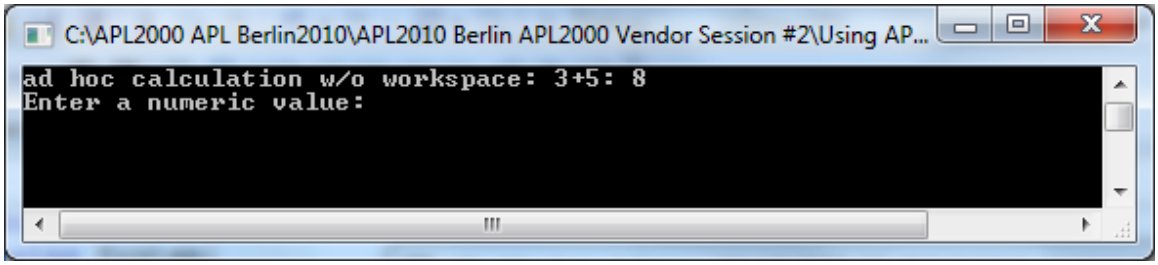
Use the Visual Studio F5 key to ‘debug’ the VisualAPL solution.



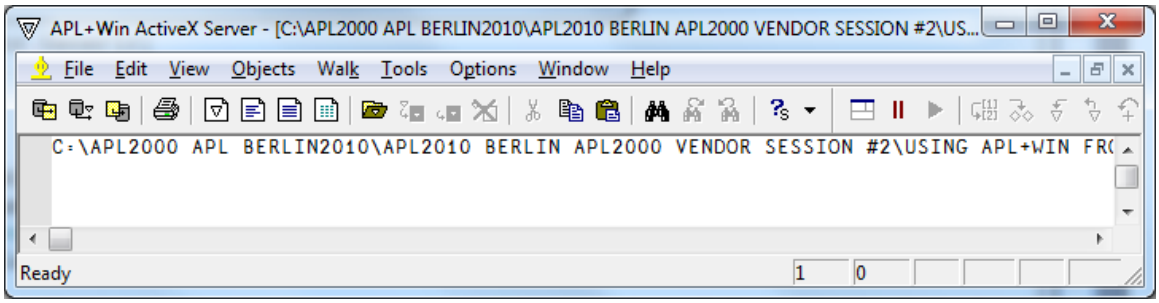

```
#region Using directives
using System;
using System.Collections.Generic;
using System.Text;
using APLW;
#endregion

namespace VAPLusingAPLWin{
    public class Program{
        public static void Main(){
        APLW.WSEngineClass apl = new APLW.WSEngineClass();
        apl.Visible=1;
        Console.WriteLine("ad hoc calculation w/o workspace: 3+5: "+ apl.Exec("3+5").ToString());
        try
        {
            string wspath = System.IO.Path.GetFullPath(@"..\..\APL+WIN\CALLAPLWINFROMVISUALAPL.W3");
            wspath = ¯3+wspath;
            apl.SysCommand("LOAD " + wspath);
            Console.WriteLine("Enter a numeric value: ");
            double input = Convert.ToDouble(Console.ReadLine());
            if (input<0)
            {
                throw new Exception("Input value is negative!");
            }
            double result = Convert.ToDouble(apl.Call("SquareRoot",input,Type.Missing));
            Console.WriteLine("Square root of "+input.ToString()+": "+result.ToString());
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
        Console.WriteLine("Click the Enter key to end this application system");
        Console.ReadLine();
        apl.Close()
        }
    }
}
```

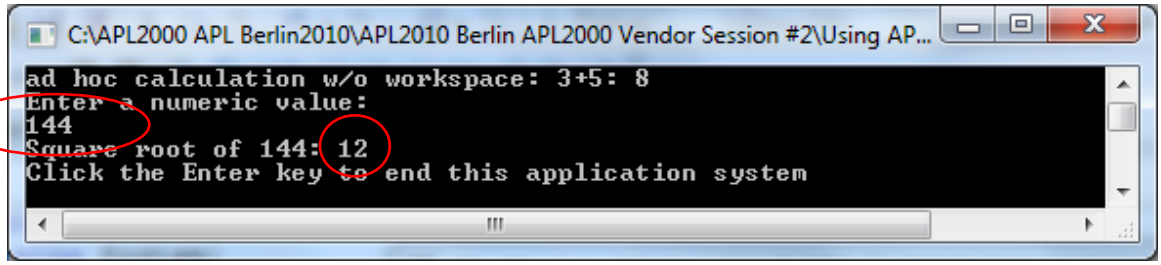
Because this is a ‘Console Project’, the command prompt window will be presented:



Because the VisualAPL project set the APL+Win Visible property to 1, the APL+Win ActiveX server window will also be presented.



Enter a numeric value into the command prompt window and click the Enter key once to cause VisualAPL to call the APL+Win ‘SquareRoot’ function.



Click the Enter key again and the VisualAPL console project will close the instance of APL+Win, the command prompt window will close ending the debug process and the Visual Studio IDE will return to programming mode.

This simple VisualAPL application system illustrates how VisualAPL can use APL+Win functions to support the business rules and calculations of an application system.

Using APL+Win in this way means that the .Net solution will not be fully-managed. However the technique outlined here can enhance the use of APL+Win especially when complex algorithms have already been implemented and validated in APL+Win. To create a fully-managed .Net solution, the APL+Win source code can be converted to VisualAPL source code. For more information on this conversion process see http://forum.apl2000.com/viewtopic.php?t=455.