TELEDYNE DALSA
Everywhereyoulook™

Developing a Visual C# .NET interface for a Sherlock 7 investigation

Overview

Although a Sherlock 7 investigation can be run from the Sherlock IDE, it is often desirable, and sometimes necessary, to hide the Sherlock GUI behind a custom interface. This tutorial walks you through the steps of creating a Visual C# .NET interface for a Sherlock 7 investigation.

Requirements

To follow the steps in this tutorial, you will need

- Sherlock 7
- Microsoft Visual C# .NET 2005 or newer[1]
- Familiarity with Visual C# .NET

[1]No claim is made for the applicability of this tutorial to earlier versions of Visual C# .NET (i.e., 2002 and 2003)

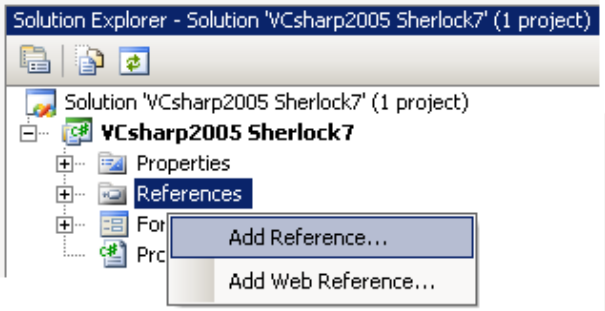
Five steps

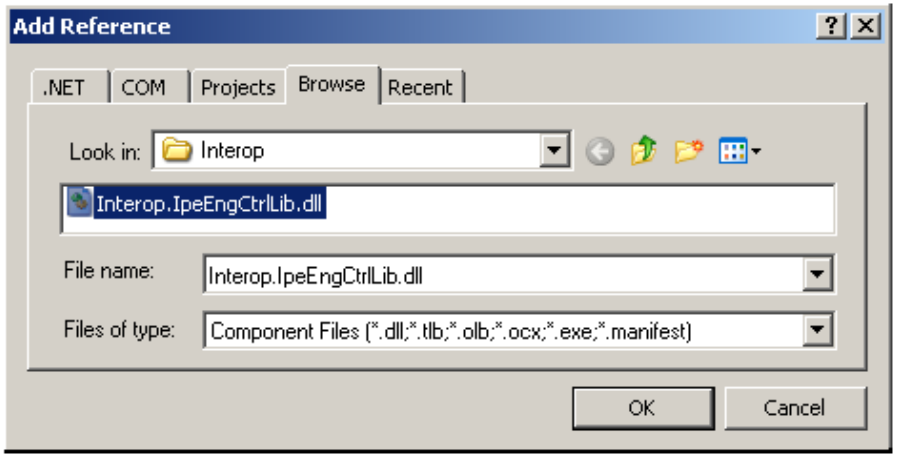
The five steps in creating a Visual C# .NET (from here on, VC#) interface for a Sherlock 7 investigation:

1. Add a reference to the Sherlock runtime engine
2. Declare and create a Sherlock object
3. Set up Sherlock display (optional, but common)
4. Use the Sherlock object to load, execute, control and communicate with an investigation
5. Destroy the Sherlock object

Add the Sherlock reference

Open a new VC# Windows Application project. In the project's **Solution Explorer** window, right-click on **References.** On the menu that pops up, click on **Add Reference…** In the **Add Reference** dialog, click the **Browse** tab, navigate to the Sherlock **Interop** directory, select **Interop.IpeEngCtrlLib.dll**, and click the **OK** button.





Create the Sherlock object

The Sherlock object must be created and initialized before you call any of its methods.

```
IpeEngCtrlLib.Engine hSherlock;
IpeEngCtrlLib.I_ENG_ERROR nErr;

public Form1(){
        InitializeComponent();

        hSherlock = new IpeEngCtrlLib.Engine();
        nErr = hSherlock.EngInitialize();
}
```

Load an investigation

The Form_Load subroutine is usually a good place to load the investigation you want to run.

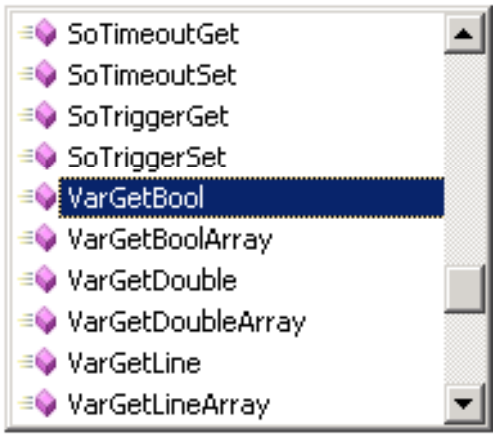
```
private void Form_Load(object sender, EventArgs e)
{
        nErr = hSherlock.InvLoad("Widget.ivs");
}
```

If you have several investigations that can use the same VC# front end (for example, different but similar parts to be analyzed), the investigation name can be contained in a variable that is filled based on user input or some other mechanism.

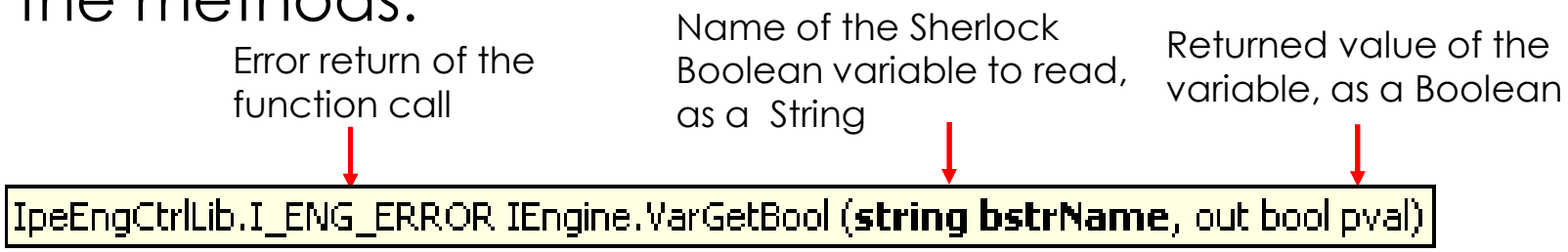
Autocomplete and pop-up tips

As you type Sherlock object code, autocomplete displays a list of the matching methods…

```
nErr = hSherlock.var
```

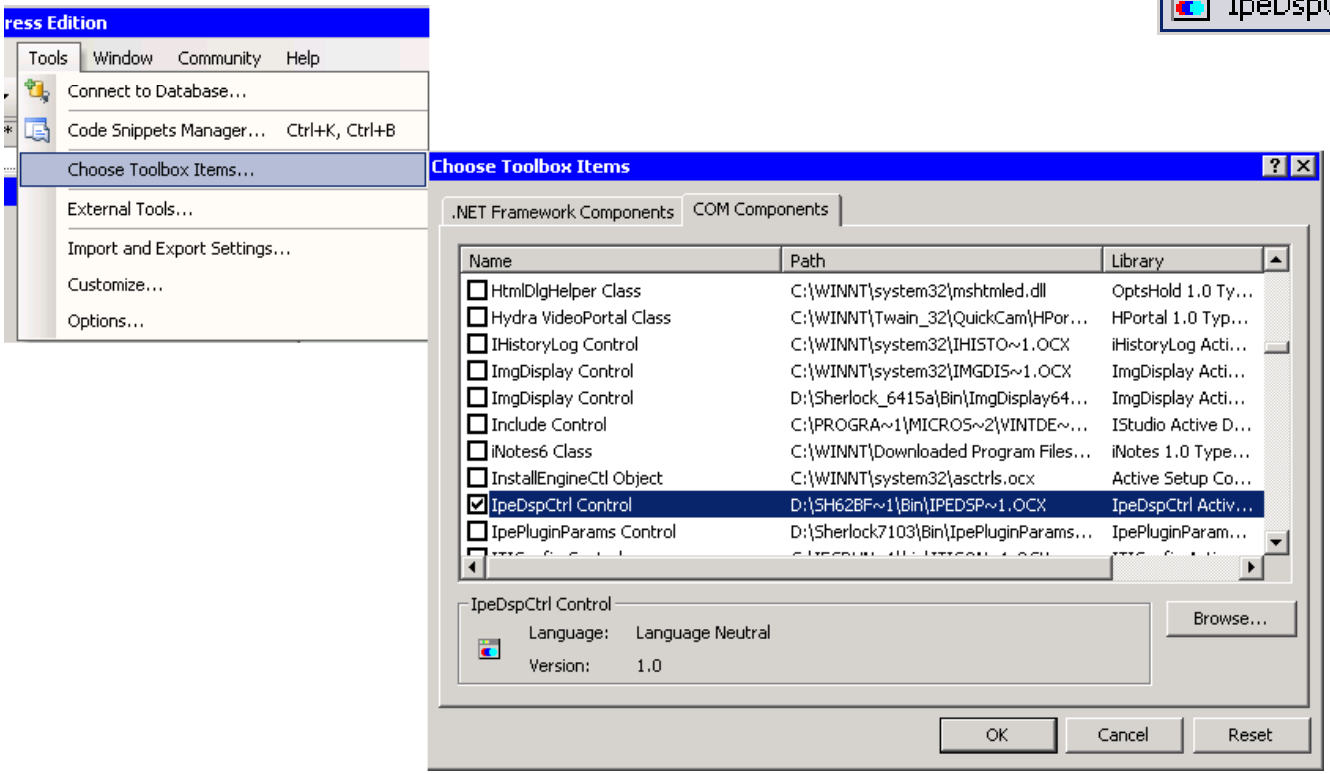


…and pop-up tips show you the required parameters for the methods.



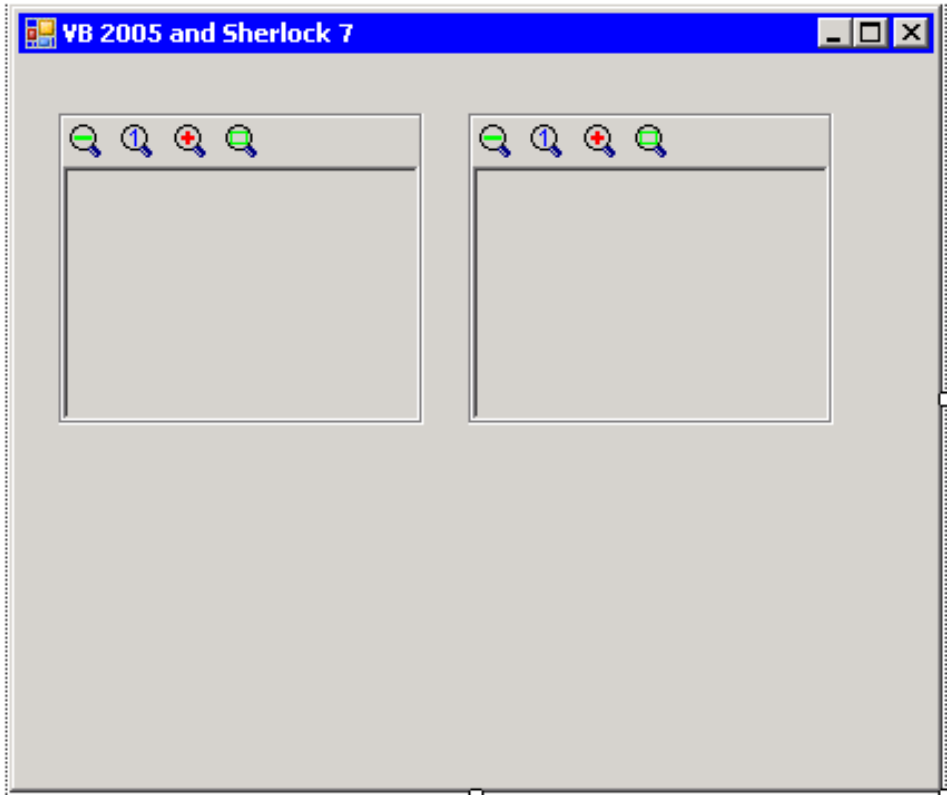
Add the IpeDspCtrl control

To display an image window on the VC# form, you must add an IpeDspCtrl display control. On the VC# Toolbox, open the **Components** tab. From the VC# project's main menu, select **Tools → Choose Toolbox Items…** On the **Choose Toolbox Items** dialog, click the **COM** Components tab, select **IpeDspCtrl Control**, and click the **OK** button. The IpeDspCtrl tool is added to the Toolbox Components tab.



Add display controls

You can add as many display controls to your VC# form as you need. Their default names are axIpeDspCtrl1, axIpeDspCtrl2, etc., but you can rename them.



Connect a display to an image window

To display an image window, you must connect it to a display control. You do not have to display any image windows, but it is common to display at least one.

```
private void Form_Load(object sender, EventArgs e)
{
        nErr = hSherlock.InvLoad("Widget.ivs");
        axIpeDspCtrl1.ConnectEngine(hSherlock.GetEngineObj());
        axIpeDspCtrl1.ConnectImgWindow("imgA");
}
```

Run the investigation

Commands to run the investigation are usually executed in response to button or menu item events.

```
private void btnRunOnce_Click(object sender, EventArgs e)
{
   // Run the investigation once
   nErr = hSherlock.InvModeSet(IpeEngCtrlLib.I_MODE.I_EXE_MODE_ONCE);
}

private void btnRunContinuously_Click(object sender, EventArgs e)
{
  // Run the investigation continuously
  nErr = hSherlock.InvModeSet(IpeEngCtrlLib.I_MODE.I_EXE_MODE_CONT);
}
```

Halt the investigation


```
private void btnHalt_Click(object sender, EventArgs e)
{
  // Halt the investigation after it finishes its current iteration
  nErr = hSherlock.InvModeSet(IpeEngCtrlLib.I_MODE.I_EXE_MODE_HALT);
}

private void btnAbort_Click(object sender, EventArgs e)
{
  // Abort the investigation immediately
   nErr = hSherlock.InvModeSet(IpeEngCtrlLib.I_MODE.I_EXE_MODE_HALT);
}
```

RunCompleted

At the end of every iteration of the investigation, the RunCompleted event is generated. You must define the run completed event handler.

```
public Main(){
  InitializeComponent();

  hSherlock = new IpeEngCtrlLib.Engine();
  nErr = hSherlock.EngInitialize();
  hSherlock.RunCompleted +=
    new IpeEngCtrlLib_IEngineEvents_RunCompletedEventHandler(
    hSherlock_RunCompleted);
}
```

RunCompleted

Add code to the RunCompleted event handler to read Sherlock algorithm readings, read and write Sherlock variables, update the VC# form, etc.

```
private void hSherlock_RunCompleted()
{
   double dblCount;

   // varCount is a Sherlock variable of type N (number)
   nErr = hSherlock.VarGetDouble("varCount", out dblCount);
   labelConnectivtyObjectCount.Text = dblCount,ToString();
        …
End Sub
```

RunCompleted

To retrieve and access a single Sherlock point:

```
IpeEngCtrlLib.I_POINT Point;
double dblX, dblY;
// varPoint is a Sherlock variable of type P (point)
nErr = hSherlock.VarGetPoint("varPoint", out Point);
dblX = Point.x;
dblY = Point.y;
```

To retrieve and access an array of Sherlock points:

```
IpeEngCtrlLib.I_POINT[] Points;
double dblX, dblY;
int intIndex;
// varPoints is a Sherlock variable of type P[ ] (array of points)
nErr = hSherlock.VarGetPointArray("varPoints", out Points);
for (intIndex = 0; intIndex < Points.length; intIndex++){
   dblX = Points[intIndex].x;
   dblY = Points[intIndex].y;
   …
}
```

RunCompleted

To retrieve and access a single Sherlock line:

```
IpeEngCtrlLib.I_LINE Line;
double dblAngle, dblDistance;
// varLine is a Sherlock variable of type L (line)
nErr = hSherlock.VarGetLine("varLine", out Line);
dblAngle = Line.a;
dblDistance = Line.d;
```

To retrieve and access an array of Sherlock lines:

```
IpeEngCtrlLib.I_LINE[] Lines;
double dblAngle, dblDistance;
int intIndex;
// varLines is a Sherlock variable of type L[ ] (array of lines)
nErr = hSherlock.VarGetLineArray("varLines", out Lines);
for (intIndex = 0; intIndex < Lines.length; intIndex++){
  dblAngle = Lines[intIndex].a;
  dblDistance = Lines[intIndex].d;
  …
}
```

Destroy the Sherlock object

It is "best practice" to destroy the Sherlock object before the VC# application terminates.

```
protected override void Dispose(bool disposing)
{
  if (disposing)
  {
    if (hSherlock != null)
    {
        hSherlock.EngTerminate();
        hSherlock = null;
    }
    if (components != null)
    {
        components.Dispose();
    }
  }
  base.Dispose(disposing);
}
```

Things to Note

1. Every call to a Sherlock object method generates a return value. Best practice is to check this value:

```
nErr = hSherlock.VarGetDouble("varCount", out dblCount);
If (nErr != IpeEngCtrlLib.I_ENG_ERROR.I_OK)
{
  //Error-handling code
}
```

2. To resize the contents of an image window to fill a display control that is either larger or smaller than the image window, call the method axIpeDspCtrl.SetZoom(0).