WINDMILL 6

IML Tools
User Manual

Windmill Software Limited

Manual Code : WM.OCX-1.2

Table of Contents

IML Tools Manual

IML Tools are for scientists and engineers creating their own measurement, control and analysis applications. You can use the Tools directly from Windows applications like Excel or Access, from JavaScript code running on a web page or when programming in Visual Basic and other languages.

The Tools extend the capabilities of your spreadsheet, database or programming language. They do this by providing methods (functions) you can include in your macros and programs.

Common measurement tasks are provided by the Windmill software suite. These ready-to-run applications provide, for example, data logging, trend charting and output control. The IML Tools come in handy when you want to use your own programs or macros to get data directly from the hardware.

Technically speaking, the IML Tools are a 32-bit Active X control (OCX). IML stands for Interface Management Language.

This manual covers:

- Installing the IML Tools;
- Using IML Tools in Visual Basic programming;
- Using IML Tools in Excel;
- Using IML Tools in JavaScript code;
- Properties and methods;
- IML Error Codes.

1 Installing the IML Tools

IML Tools run alongside Version 6 of Windmill, under Windows 2000, NT, 98 or 95. Install Windows and Windmill before IML Tools.

You may have been supplied with IML Tools on floppy disk, or been e-mailed a zip file.

Installing from Disk

1. From the Windows Start menu select Run.
2. Type d:\setup and press OK. The Tools are copied to the windows\system folder.
3. Shut down Windows and restart your computer.
4. Create a file to configure your hardware---see Section 1.1.

Installing from E-mailed Files

1. Unzip the files into a temporary folder.
2. From the Windows Start menu select Run.
3. Type c:\folder name\setup and press OK. The Tools are copied to the windows\system folder.
4. Shut down Windows and restart your computer.
5. Create a file to configure your hardware---see Section 1.1.

1.1 Creating a File to Configure your Hardware

You've now installed IML Tools and can use them in your programs. Before you do so, though, use the Windmill applications to create one or more hardware configuration files. You can then configure the hardware from your own programs simply by loading one of these files.

To create a file

1. Run Windmill ConfIML to add and configure device drivers. See the *Windmill IML Installation and Configuration Manual*, or the ConfIML Help file, for details.
2. Run Windmill SetupIML. This lets you choose how you wish to use the hardware. It scans the devices you added in ConfIML and builds a default set-up file. You can edit the

set-up (selecting the measurement type, data format, alarms, engineering units, etc) and store libraries of files. All your programs, macros and scripts can then configure the hardware with just one line of code.

For your first set-up file it's probably best to use just the Software Signal Generator, and ignore your hardware for the time being. The Signal Generator mimics a hardware device driver----so you can practice using Windmill and the IML Tools safe in the knowledge that any problems you encounter are not caused by the hardware.

See the Chapter 3 of the *Windmill User Manual*, or the SetupIML Help file, for details of using SetupIML.

2 Using IML Tools in Visual Basic Programming

1. Start a project in Visual Basic 5 or later.
2. From the Projects menu select Components.
3. Check the IMLtools.IMLcontol box. A user control icon appears in your VB toolbox bar.
4. Place the control on your form and use its properties and methods in your code. The Windmill icon appears as the control image and the default name is IMLcontrol1.

2.1 Components of the IML VB Code: Initialisation, Reading & Writing, and Ending

To get you started, here are examples of code you'd insert in the beginning, middle and end of your data acquisition program. For a more detailed sample code see the next section. For a full description of the IML Tools properties and methods see Section 5.

Initialisation

At the beginning of your program you need to open the IML Tools control and configure the hardware.

```
ret%=IMLcontrol1.IMLOpen
ret%=IMLcontrol1.IMSloadfile(IMSfilespec$)
```

IMSfilespec$ is the full path and name of the hardware setup file you created with the Windmill SetupIML application (Section 1.1). You might also like to check that ret% = true for both calls, verifying no errors have occurred.

Reading & Writing

In the body of your program you can read data from your hardware, and send data to your hardware.
To read a numeric value use

```
ret%=IMLcontrol1.ReadFromChannel(myChannel-
Name$,sVAl$)
```

```
xVal=val(sVal$)
```

To send a data string (eg "1.234") to an output channel use

```
sVal$="1.234"
ret%=IMLcontrol1.SendToChannel(myChannel-
Name$,sVAl$)
```

The channel names are those you chose in SetupIML and saved in your setup file.

Ending

Before ending your program, for example in your Form_Unload subroutine, use

```
ret%=IMLcontrol1.IMLClose
```

2.2 Sample Code for a VB Application

This example:

- Displays the readings from 2 analogue inputs, taken once per second, in label controls.
- A text control allows the value of an analogue (or digital) output channel to be varied.
- The value on the analogue output is changed when the text in the control is changed.

The VB form uses the following controls:

Timer	name: Timer1
	enabled: true
	interval: 1000
Labels	name: laAnIn1
	name: laAnIn2
Text box	name: text1
IMLcontrol	name: IMLcontrol1

Add the IMLcontrol to your project using the Project.Components option on the VB menu bar

VB Code

```
Private Sub Form_Load()
' Requires an *.ims file (created in SetupIML) with
' the following channels
' AnIn1: enabled for input
' AnIn2: enabled for input
' and optionally:
' AnOut1: enabled for input and output

x% = IMLcontrol1.IMLOpen
' Load the IMS file to set up the hardware
IMSfileSpec$ = "c:\windmill\testctrl.ims"
x% = IMLcontrol1.IMSLoadFile(IMSfileSpec$, a$)
' Show the IMS setup name on the form caption
Caption = a$

End Sub

Private Sub Form_Unload(Cancel As Integer)
' Close down this link to the IML libraries
x% = IMLcontrol1.IMLclose
End Sub

Private Sub Text1_Change()
' When the value in this text box changes send the
' string to the output channel
' Note this requires an Analogue or Digital Output
' channel on your hardware
x% = IMLcontrol1.SendToChannel("AnOut1", Text1.Text)
End Sub

Private Sub Timer1_Timer()
' Enable Timer1 to tick every 1000 msecs
' Read two channels and display them on the screen
x% = IMLcontrol1.ReadFromChannel("AnIn1", a$)
laAnIN1.Caption = a$
x% = IMLcontrol1.ReadFromChannel("AnIn2", a$)
laAnIn2.Caption = a$
End Sub
```

3 Using IML Tools in Excel

The Tools help you create macros using Excel's VBA Editor. The following steps explain how to place a button on your spreadsheet that reads data every time you click it. The data comes from a channel on the Software Signal Generator and is shown in one of the cells of the spreadsheet.

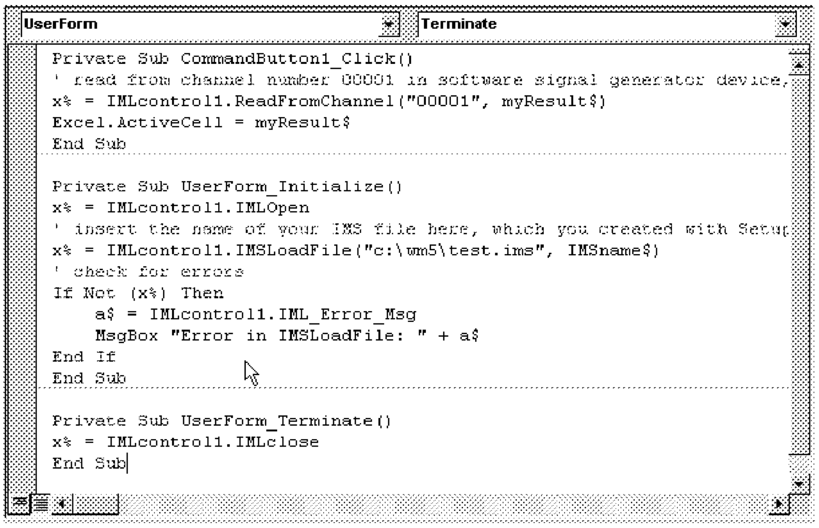
The Software Signal Generator simulates a device with seven channels. It lets you experiment with the IML Tools whilst eliminating the hardware as a source of problems.

1. Load Excel with a clean sheet. Select the cell in which you want to display the data.
2. In Excel 97 and later: from the Tools menu select Macro and choose Visual Basic Editor. (In Excel Versions 5 to 7: from the Insert menu select Macro and choose Module.)
3. Create a form: select User Form from the Insert menu.
4. Enable the IML Tools. From the Tools menu select Additional Controls. Scroll through this list until you find IMLtools.IMLcontol and check its box.
5. In the Toolbox select the IML Tools icon.
6. Place the IML Tools control on the form by dragging a rectangular outline with the mouse. A Windmill logo shows that the Tools have been successfully added to the form.
7. You now need to add the control button which will be shown over the spreadsheet. In the Toolbox select the control button icon, then click the form to place it there.
8. Choose the text that will be shown on the button. In the Properties box find the Caption and type a label for the button - "Get Data" for example.
9. You're now ready to enter the code to read the data. Double-click the form and enter the code shown overleaf. We've also placed the code in a file on our web site so you

can copy and paste it into the form. Go to http://www.windmill.co.uk/imltools/imltexcl.txt (A full explanation of the properties and methods available with IML Tools is given in Section 5.)

10. Run the code. From the Run menu select Run Sub/User Form.
11. In the spreadsheet press the Get Data button. The spreadsheet will take a reading and update its active cell.
12. Close the form before closing the spreadsheet.

```
Private Sub CommandButton1_Click()
' read from channel number 00001 in software signal
' generator device, and store data in myResults$
x% = IMLcontrol1.ReadFromChannel("00001", myResult$)
Excel.ActiveCell = myResult$
End Sub

Private Sub UserForm_Initialize()
x% = IMLcontrol1.IMLOpen
' insert the name of your IMS file here,
' which you created with SetupIML
x% = IMLcontrol1.IMSLoadFile("c:\wm5\test.ims",
IMSname$)
' check for errors
If Not (x%) Then
    a$ = IMLcontrol1.IML_Error_Msg
    MsgBox "Error in IMSLoadFile: " + a$
End If
End Sub

Private Sub UserForm_Terminate()
x% = IMLcontrol1.IMLclose
End Sub
```



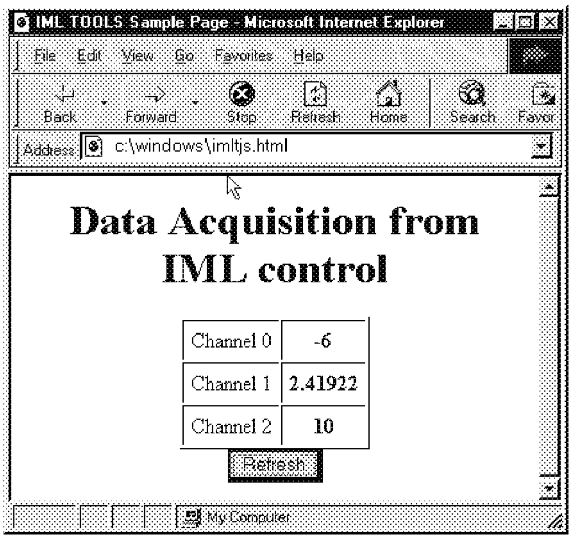
```
Private Sub CommandButton1_Click()
' read from channel number 00001 in software signal generator device,
x% = IMLcontrol1.ReadFromChannel("00001", myResult$)
Excel.ActiveCell = myResult$
End Sub

Private Sub UserForm_Initialize()
x% = IMLcontrol1.IMLOpen
' insert the name of your IMS file here, which you created with Setup
x% = IMLcontrol1.IMSLoadFile("c:\wm5\test.ims", IMSname$)
' check for errors
If Not (x%) Then
    a$ = IMLcontrol1.IML_Error_Msg
    MsgBox "Error in IMSLoadFile: " + a$
End If
End Sub

Private Sub UserForm_Terminate()
x% = IMLcontrol1.IMLclose
End Sub
```

In this section we've shown you how to read data from one "hardware" channel and display it in Excel. You can do much more with the IML Tools---for full details of the properties and methods available turn to Section 5. For more information on creating macros with Excel's Visual Basic Editor please refer to the Excel Help or Manual.

4 Using IML Tools in JavaScript Code

You can use the IML Tools in JavaScript code on a web page. We've put the example given here on our web site at http://www.windmill.co.uk/imltools/jscript.zip - so you can download the code rather than typing it. The script will only work with browsers that support Active X such as Microsoft Internet Explorer.

4.1 Example HTML using JavaScript to Read Data and Display it in a Web Page


IML TOOLS Sample Page - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favor
Address c:\windows\imltjs.html
Refresh
My Computer

Data Acquisition from IML control

Channel 0	-6
Channel 1	2.41922
Channel 2	10

```
<HTML>
<HEAD>
<TITLE>IML TOOLS Sample Page</TITLE>

<OBJECT ID="IMLcontrol" WIDTH=64 HEIGHT=63
 CLASSID="CLSID:1FCF5582-2A14-11D3-9743-525400DDF830"
 CODEBASE="http://www.domainname.co.uk/imltools.ocx">
    <PARAM NAME="_ExtentX" VALUE="1693">
    <PARAM NAME="_ExtentY" VALUE="1667">
</OBJECT>
```

```
<SCRIPT LANGUAGE="JavaScript">

// This example uses 3 input channels from the IML
// device. The names of the channels are placed in
the
// HTML element where the result will go. The data
// values will be posted onto the form in the element
// with ID in ChanXtxt
var Chan1Name="";
var Chan2Name="";
var Chan3Name="";
var Chan1txt="";
var Chan2txt="";
var Chan3txt="";

function IMLreadChannel(xChan, xText)
{
   var sVal="abc";
   var x;
   // Uses the windmill IML control to read from a

   // single input channel
   x=IMLcontrol.ReadFromChannel(xChan, sVal);
   if (x==0)
   {  alert("Unable to read from IMS channel: "+xChan);
      sVal="n.a.";
   }
   else
   {  sVal=IMLcontrol.IML_LastDataString;
   }
   // Update the text in the browser document
   xText.innerText=sVal;
}

function getIMLdata()
{
   var x=0;
   // This is the handler for the refresh button push
   x=IMLreadChannel(Chan1Name, Chan1txt);
   x=IMLreadChannel(Chan2Name, Chan2txt);
   x=IMLreadChannel(Chan3Name, Chan3txt);
}
```

```
// called at body onLoad
function initQuestion()
{
   var IMSname="abc";
// note - the first \ character is ignored, but al-
lows
// the second to be recognised as the actual \ in the
// file specification, so this will go through
// to the IMLtools control as "c:\wm5\test.ims"
   var IMSfilespec="c:\\wm5\\test.ims";
   var x=0;
   x=IMLcontrol.IMLOpen();
   x=IMLcontrol.IMSLoadFile(IMSfilespec,IMSname);
// note - IMSname is not returned from the control

// into JavaScript
   if (x==0)
   {  alert("Unable to load IMS file: "+IMSfilespec);
   }
// get the names of the IML channels to read, and the
// ID of the table elements where the data values
will
// be posted
   Chan1txt=document.all("CHAN0");
   Chan1Name=Chan1txt.innerText;
   Chan2txt=document.all("CHAN1");
   Chan2Name=Chan2txt.innerText;
   Chan3txt=document.all("CHAN2");
   Chan3Name=Chan3txt.innerText;
   x=getIMLdata();
}

</SCRIPT>
</HEAD>

<BODY bgcolor="lemonchiffon" onLoad="initQuestion()">
<H1><CENTER>Data Acquisition from IML control<font
color ="black"></CENTER></H1>
<FORM NAME="control">
<TABLE CELLPADDING="5" CELLSPACING="1" BORDER="1"
ALIGN="CENTER">
<font color ="darkslateblue">
```

```
<TR ALIGN="CENTER">
<TD>Channel 0
</TD>
<TD><SPAN ID="CHAN0">00000</SPAN>
</TD>
</TR>

<TR ALIGN="CENTER">
<TD>Channel 1
</TD>
<TD><SPAN ID="CHAN1">00001</SPAN>
</TD>
</TR>

<TR ALIGN="CENTER">
<TD>Channel 2
</TD>
<TD><SPAN ID="CHAN2">00002</SPAN>
</TD>
</TR>

</TABLE>
<CENTER>
<P>
<INPUT TYPE=button VALUE="Refresh" NAME="pbRefresh"
onclick="getIMLdata()">
<P>
</CENTER>
</FORM>
</BODY>
</HTML>
```

4.2 Notes on the JavaScript Example

CLASSID="CLSID:1FCF5582-2A14-11D3-9743-525400DDF830"

The CLASSID identifies the IML Tools. This will change with different versions of the Tools. To find your CLASSID from the Windows Start menu select Run. Enter regedit and press OK. You should find imltools.IMLtoolsv6 in MyComputer\HKEY_CLSSES_ROOT_CLSID\

CODEBASE="http://www.mydomain.co.uk/imltools.ocx">

Should the Browser not be able to find the IML Tools on the computer on which it is running, it will download them from the path specified in the CODEBASE. Please remember though that unless a multiple site licence has been purchased, each copy of the IML Tools can only be used with real hardware on one computer.

Trouble Shooting

Should you have problems with your code it's good practice to close the IML Driver before retesting. Look for the IML Device icon at the bottom of the screen, right click and select Close.

When you open your web page in the browser you might see this message:

```
Security Alert:
An ActiveX object on this page may be unsafe
...
```

Click Yes to continue and allow the control to be initialised.

You can avoid the message by marking the control as safe for initialisation and scripting. To do this edit the Windows Registry.

From the Windows Start menu select Run and type regedit. The Registry Editor appears.

In the Registry Editor, select Find from the Edit menu and type imltools. If the IMLtools have been registered the search will find the CLSID: 1FCF5582..., marked by an open folder. Click the Implemented Categories sub-folder.

Within Implemented Categories there are normally 4 folders. You need to add 2 more. With the Implemented Categories folder selected, select New from the Edit menu and choose Key. A folder is created called NewKey #1. Rename this as

{7DD95802-9882-11CF-9FA9-00AA006C42C4}
this marks the control as safe for initialisation.

Repeat the process to add the key:
{7DD95801-9882-11CF-9FA9-00AA006C42C4}
this marks the control as safe for scripting.

Your web page will now open without the Security Alert.

5 Properties and Methods

The IML Tools are used in *object-orientated programming*. The Visual Basic, Excel and JavaScript examples given all view the IML Tools control as an object. Like other objects, the IML Tools has its own properties and methods.

Properties tell you something about an object: they describe some aspect of it. Methods on the other hand are something you do with an object. You could think of properties as being adjectives, methods as being verbs and objects as being nouns.

The properties of the IML Tools control tell you something about its last action. The methods tell the IML Tools to do something: open communications with the IML Tools library, configure the hardware by sending it a setup file, take a reading from an instrument, and so on.

5.1 Properties of the IML Tools

`.ErrorCode`

The error code generated by the last operation. Returns 0 for no error. For details of error codes see Section 6.

`.ErrorString`

A string representation of the error generated by the last operation---ie an explanation that people can understand. Returns Normal for no error.

`.LastDataString`

The last data string received when using the ReadFromChannel method. Used in JavaScript when the datastring is not returned in the function call parameter.

5.2 Methods for the IML Tools

These all return a true (-1) or false (0) result in an integer format.

`.IMLOpen`

Opens a link to the IML libraries, and if the drivers specified in ConfIML are not already loaded, loads them.

`.IMSLoadFile(Filespec)`

Loads the specified ims file (as created in the Windmill SetupIML application) into the system, thus configuring the hardware to your specifications. Use this after the “IMLOpen” method.

`.IMSName`

Returns the name of the setup file (not used in JavaScript).

`.ReadFromChannel (ChannelName, DataString)`

Reads a string from the specified channel and returns the reading in DataString.The channel name is that you chose in SetupIML and saved in your setup file. It is also a string.

`.SendToChannel (ChannelName, DataString)`

Sends the DataString to the specified channel. For a number, use the string representation of that number. For example “3.12". The channel name is that you chose in SetupIML and saved in your setup file. It is also a string.

`.RezeroChannel (ChannelName)`

Resets the Offset parameter for the engineering units calculation, so that the current reading from the input channel will be returned as 0.0. Subsequent readings are returned as relative to this value. The channel name is that you chose in SetupIML and saved in your setup file. It is also a string.

This is useful, for example, when you want to zero a balance before taking further measurements. It is also useful when the change to be measured is very small compared to the measurement value----in strain applications for example.

`.IMLClose`

Releases the link to the IML libraries. Use this at the end of your code.

6 Error Codes

An error code of zero indicates that the command was successfully completed. Any other error code indicates an error has occurred. The error condition is cleared (and the code is no longer available) when any other command is executed. The subsequent command may set an error condition of its own.

Here is a list of IML error codes and their meanings. Unused numbers are reserved for future use.

100	Unrecognised command or macro
101	Unspecified hardware failure
102	Unspecified software failure
103	Insufficient memory
104	Hardware module failed to accept data
105	Hardware module failed to supply data
106	Unable to configure Hardware module
107	Error in communication from host
108	Input value out of range
109	Output value out of range
110	Insufficient memory to store macro
111	Internal memory corrupted
112	Macros nested too deeply, or recursive macro
113	Hardware set-up not suitable
114	Data not yet ready
115	Non-volatile storage corrupted
116	Unable to write to non-volatile storage
117	Unable to trigger high speed scan

310	Channel number not available
311	Channel name not known
312	Range of channels incorrectly specified
313	Channel is not enabled, or is grouped
314	Channel does not support reading
320	Channel number not available
321	Channel name not known
322	Channel cannot accept output data
323	Channel is not enabled, or is grouped

324	value is missing, or in wrong format
390	Module number not acceptable
391	Byte count not acceptable
392	Trigger flag value not acceptable