ASCOM Telescope v3 sample app in C++/MFC, VS2013

by Paul Kanevsky

Simple example showing how to use C++/MFC to connect to an ASCOM Telescope driver. Includes chooser, connect/disconnect, RA/DEC display and RA Tracking rates list.

Contains convfiguration for 32- and 64-bit versions.

Feel free to modify/add/enhance in any way you see fit.

———————————

Usage Notes

If you are building an ITelescope v3 app, I suggest you simply copy the following .h files into your own project, instead of trying to generate them using MFC Class Wizard. These have been modified from their original form to ensure that this works properly with drivers that have different DISPIDs:

CAxisRates.h
Chooser.h
CRate.h
CTelescopeV3.h
CTrackingRates.h

You can then start using these classes directly in your own app. See ASCOMTeleDlg.cpp file for example:

```
CChooser  ch;
COleException err;

if (!ch.CreateDispatch(L"ASCOM.Utilities.Chooser", &err)) return;

ch.put_DeviceType(L"Telescope");
CString s =  ch.Choose(m_sTelescopeID);
```

For those who want to start from scratch or to use other ASCOM interfaces, here are the critical steps needed to re-create this sample app:

1. Create an MFC project
2. Add /clr option to support Common Language Runtime (under project Configuration Properties)
3. Invoke MFC Class Wizard to add new class (Project->Add Class menu)
4. Select MFC Class From TypeLib
5. Select From File, and click on … to browse file system
6. Navigate to C:\Windows\assembly\GAC_MSIL\ASCOM.DeviceInterfaces and select the folder in there that starts with your platform version (6.0.0.xxxx)
7. Select ASCOM.DeviceInterfaces.tlb file in that folder
8. Now choose the interfaces you want to add to your project from the left panel, and add them to the right panel using the '>' or '>>' button
9. Confirm. This will add automatically generated .h files for all the classes you wanted to add.

Now, the fun part is to edit the .h files to ensure that the DISPIDs used for all the methods and properties are not hard-coded. To do this, you'll want to copy and paste my CACHE_DISPID macro definition into each .h file, near the top:

```
#define CACHE_DISPID(name) \
      static DISPID id;\
      if (id == NULL) {\
      CString ss = (name);\
      BSTR szMember = ss.AllocSysString();\
      m_lpDispatch->GetIDsOfNames(IID_NULL, &szMember, 1,
LOCALE_USER_DEFAULT, &id);\
      SysFreeString(szMember);\
      }
```

For each COM property or method call in the .h file, you'll want to use this macro at the beginning of the function, and change InvokeHelper to pass 'id' instead of hardcoded DISPID:

```
	BOOL get_Connected()
	{
		CACHE_DISPID(L"Connected");
		BOOL result;
		InvokeHelper(id, DISPATCH_PROPERTYGET, VT_BOOL, (void*)&result,
NULL);
		return result;
	}
```

That's it! You may want to comment out the #import directive near the top of the .h file(s) to stop MFC from re-generating the include file, as this will overwrite the changes you just made above.