Determining how each Explorer window is sorted

devblogs.microsoft.com/oldnewthing/20151228-00

December 28, 2015

Raymond Chen

Today’s Little Program lists all the open Explorer windows and their current sort criteria. (I will refrain from mentioning how auto-sort makes this meaningful.) Remember that Little Programs do little to no error checking.

```
#define STRICT
#include <windows.h>
#include <ole2.h>
#include <shlobj.h>
#include <shdispid.h>
#include <atlbase.h>
#include <atlalloc.h>
#include <stdio.h>

void ProcessOneWindow(IUnknown *punk)
{
 CComPtr<IShellBrowser> spsb;
 if (FAILED(IUnknown_QueryService(punk, SID_STopLevelBrowser,
                                  IID_PPV_ARGS(&spsb)))) return;

 CComPtr<IShellView> spsv;
 if (FAILED(spsb->QueryActiveShellView(&spsv))) return;

 CComQIPtr<IFolderView2> spfv(spsv);
 if (!spfv) return;

 CComHeapPtr<WCHAR> spszLocation;
 if (FAILED(GetLocationFromView(spsb, &spszLocation))) return;

 printf("Location = %ls\n", static_cast<PCWSTR>(spszLocation));

 int cColumns;
 if (FAILED(spfv->GetSortColumnCount(&cColumns))) return;
 if (cColumns > 10) cColumns = 10;

 SORTCOLUMN rgColumns[10]; // arbitrary number
 spfv->GetSortColumns(rgColumns, cColumns);

 for (int i = 0; i < cColumns; i++) {
  PCWSTR pszDir = rgColumns[0].direction > 0 ? L"ascending"
                                             : L"descending";
  PCWSTR pszName;
  CComHeapPtr<WCHAR> spszName;
  WCHAR szName[PKEYSTR_MAX];
  if (SUCCEEDED(PSGetNameFromPropertyKey(rgColumns[0].propkey,
                                         &spszName))) {
   pszName = spszName;
  } else {
   PSStringFromPropertyKey(rgColumns[0].propkey,
                           szName, ARRAYSIZE(szName));
   pszName = szName;
  }
  printf("Column = %ls, direction = %ls\n", pszName, pszDir);
 }
}

int __cdecl wmain(int, wchar_t **)
```

```
{
 CCoInitialize init;
 CComPtr<IShellWindows> spShellWindows;
 spShellWindows.CoCreateInstance(CLSID_ShellWindows);

 CComPtr<IUnknown> spunkEnum;
 spShellWindows->_NewEnum(&spunkEnum);
 CComQIPtr<IEnumVARIANT> spev(spunkEnum);

 for (CComVariant svar;
      spev->Next(1, &svar, nullptr) == S_OK;
      svar.Clear()) {
  ProcessOneWindow(svar.pdispVal);
 }
 return 0;
}
```

To process a window, we first ask for the top-level browser, and from that we ask for the active shell view, then convert it to an `IFolderView2`. We ask for the 2 because that's the one that lets us query sort columns.

If anything goes wrong up to this point, it's probably because the window doesn't support sorting, so we won't bother printing it.

We print the location of the window using a helper function from a long time ago.

Now the interesting part: We ask for the number of sort columns, then ask for those columns.

That's it. The rest is boring again: We print each of the sort columns and the sort direction.

The main program loops through all the open Shell windows (which includes both Explorer and Internet Explorer) and processes each one.

Raymond Chen

Follow