Obtaining information about the user's wallpaper on multiple monitors

devblogs.microsoft.com/oldnewthing/20140602-00

June 2, 2014

Raymond Chen

Today we're going to dump information about the user's wallpaper settings on multiple monitors.

The idea is simple. You use the `IDesktopWallpaper` interface on the `DesktopWallpaper` object to get information about the desktop wallpaper. It will tell you the wallpaper positioning information, whether a single image is being used for all monitors, where those monitors are, and which image is on which monitor.

```
#define UNICODE
#define _UNICODE
#define STRICT
#include <windows.h>
#include <shlobj.h>
#include <atlbase.h>
#include <atlalloc.h>
#include <stdio.h> // horrors! mixing C and C++!
int __cdecl wmain(int, wchar_t **)
{
 CCoInitialize init;
 // Create the DesktopWallpaper object
 CComPtr<IDesktopWallpaper> spdw;
 CoCreateInstance(CLSID_DesktopWallpaper, nullptr, CLSCTX_ALL,
                  IID_PPV_ARGS(&spdw));
 // See if there is a single wallpaper on all monitors.
 CComHeapPtr<wchar_t> spszCommonWallpaper;
 HRESULT hr = spdw->GetWallpaper(nullptr, &spszCommonWallpaper);
 switch (hr) {
 case S_OK:
  printf("Same wallpaper on all monitors: %ls\n",
         static_cast<wchar_t *>(spszCommonWallpaper));
  break;
 case S_FALSE:
  printf("Different wallpaper on each monitor\n");
  break;
 default:
  printf("Mysterious error: 0x%08x\n", hr);
  break;
 }
 // Get the number of monitors,
 UINT count;
 spdw->GetMonitorDevicePathCount(&count);
 printf("There are %d monitors\n", count);
 // Print information about each monitor.
 for (UINT i = 0; i < count; i++) {
  // Get the device path for the monitor.
  CComHeapPtr<wchar_t> spszId;
  spdw->GetMonitorDevicePathAt(i, &spszId);
  printf("path[%d] = %ls\n",
         i, static_cast<wchar_t *>(spszId));
  // Get the monitor location.
  RECT rc;
  spdw->GetMonitorRECT(spszId, &rc);
  printf("rect = (%d, %d, %d, %d)\n",
         rc.left, rc.top, rc.bottom, rc.right);
  // Get the wallpaper on that monitor.
  CComHeapPtr<wchar_t> spszWallpaper;
  hr = spdw->GetWallpaper(spszId, &spszWallpaper);
  printf("image = %ls\n",
         static_cast<wchar_t *>(spszWallpaper));
 }
```

```
  return 0;
}
```

The program proceeds in a few basic steps.

We create the `DesktopWallpaper` object. That object will give us the answers to our questions.

Our first question is, “Is the same wallpaper being shown on all monitors?” To determine that, we call `IDesktopWallpaper::GetWallpaper` and specify `nullptr` as the monitor ID. The call succeeds with `S_OK` if the same wallpaper is shown on all monitors (in which case the shared wallpaper is returned). It succeeds with `S_FALSE` if each monitor has a different wallpaper.

To get information about the wallpaper on each monitor, we iterate through them, first asking for the monitor device path, since that is how the `DesktopWallpaper` object identifies monitors. For each monitor, we ask for its location and the wallpaper for that monitor. Note that if the monitor is not displaying a wallpaper at all, the `GetWallpaper` method succeeds but returns an empty string.

And that’s it. You can juice up this program by asking for wallpaper positioning information, and if you are feeling really adventuresome, you can use the `SetWallpaper` method to change the wallpaper.

Raymond Chen

Follow