How can I add an environment variable to a process launched via ShellExecuteEx or IContextMenu?

devblogs.microsoft.com/oldnewthing/20240131-00

January 31, 2024

Raymond Chen

The `ShellExecuteEx` function and `IContextMenu` interface provide the caller a number of places to customize how the execution occurs by allowing the call to pass a "site" which `ShellExecuteEx`/`IContextMenu` will access at various points in the execution process.

Today we'll demonstrate this technique by adding environment variables to processes launched via `ShellExecuteEx` and `IContextMenu`.

The `ICreatingProcess` interface is used by `ShellExecuteEx`. and `IContextMenu` to allow the caller to customize how processes are created. The extension point is obtained by querying the site for `SID_ExecuteCreatingProcess` and requesting an `ICreatingProcess`. If one is produced, then the system calls the `OnCreating` method with an object that allows the creation to be customized.

Today's C++ COM library is (rolls dice) C++/WinRT.

```
struct AddEnvironmentVariableSite :
    winrt::implements<AddEnvironmentVariableSite,
        ::IServiceProvider,
        ::ICreatingProcess>
{
    IFACEMETHOD(QueryService)
        (REFGUID service, REFIID riid, void** ppv)
    {
        if (service == SID_ExecuteCreatingProcess)
        {
            return this->QueryInterface(riid, ppv);
        }
        else
        {
            *ppv = nullptr;
            return E_NOTIMPL;
        }
    }

    IFACEMETHOD(OnCreating)(ICreateProcessInputs* inputs)
    {
        return inputs->SetEnvironmentVariable(
            L"EXTRAVARIABLE", L"Bonus");
    }
};
```

This site responds to `SID_ExecuteCreatingProcess` by returning itself, and it implements `ICreatingProcess` by having its `OnCreating` method set an environment variable called `EXTRAVARIABLE` with a value of `Bonus`. Since that is the only thing we do, we can just return the result of `SetEnvironmentVariable()` as our own return value. If you intend to add multiple environment variables, you would check the return value of each call to `SetEnvironment-Variable()`.

In general, your custom site would be part of a so-called “site chain” and forward any unhandled services to your own site, so that an outer site could respond to it.

Here’s an example of how to use this special environment variable site in conjunction with `ShellExecuteEx`:

```
BOOL Sample()
{
    SHELLEXECUTEINFO sei{ sizeof(sei) };
    sei.lpFile = LR"(C:\Windows\system32\charmap.exe)";
    sei.nShow = SW_SHOWNORMAL;
    auto site = winrt::make_self<AddEnvironmentVariableSite>();
    sei.hInstApp = reinterpret_cast<HINSTANCE>(site.get());
    sei.fMask = SEE_MASK_FLAG_HINST_IS_SITE;
    return ShellExecuteEx(&sei);
}
```

To pass a site to `ShellExecuteEx`, we put it in the `hInstApp` member and set the `SEE_MASK_FLAG_HINST_IS_SITE` flag to say "If you look at the `hInstApp`, you'll find a site!"[1]

For context menus, we explicitly set our custom site as the context menu's site. Building on our original sample for hosting an `IContextMenu`:

```
void OnContextMenu(HWND hwnd, HWND hwndContext, UINT xPos, UINT yPos)
{
  IContextMenu *pcm;
  if (SUCCEEDED(GetUIObjectOfFile(hwnd, L"C:\\Windows\\clock.avi",
                  IID_IContextMenu, (void**)&pcm))) {
    HMENU hmenu = CreatePopupMenu();
    if (hmenu) {
      if (SUCCEEDED(pcm->QueryContextMenu(hmenu, 0,
                            SCRATCH_QCM_FIRST, SCRATCH_QCM_LAST,
                            CMF_NORMAL))) {
        CMINVOKECOMMANDINFO info = { 0 };
        info.cbSize = sizeof(info);
        info.hwnd = hwnd;
        info.lpVerb = "play";
        auto site = winrt::make_self<AddEnvironmentVariableSite>();
        IUnknown_SetSite(pcm, site.get());
        pcm->InvokeCommand(&info);
        IUnknown_SetSite(pcm, nullptr);
      }
      DestroyMenu(hmenu);
    }
    pcm->Release();
  }
}
```

(I'm ignoring the fact that `winrt::make_self` can throw an exception which results in a memory leak in the sample code above.)

[1] This is admitted a rather ugly way to pass a site, but the ability to add a site was retrofitted onto the existing structure, so we had to hide it in an other-wise unused input member.