Demonstrating what happens when a parent and child window have different UI states

devblogs.microsoft.com/oldnewthing/20171123-00

November 23, 2017

Raymond Chen

Last time, we dug into the statement in the documentation that says, "When you change the parent of a window, you should synchronize the UISTATE of both windows." Today we'll set up a situation where the states are out of sync so we can play with it.

Start with the new scratch program and make these changes:

```
HWND g_hwndPotato;

LRESULT RootWindow::OnCreate()
{
 g_hwndChild = CreateWindow(L"Button", L"&Reset",
  WS_CHILD | WS_VISIBLE | WS_TABSTOP | BS_PUSHBUTTON,
  0, 0, 100, 100, m_hwnd, (HMENU)1, g_hinst, 0);
 if (!g_hwndPotato) {
   g_hwndPotato = CreateWindow(L"Button", L"&Potato",
    WS_CHILD | WS_VISIBLE | WS_TABSTOP | BS_PUSHBUTTON,
    0, 100, 100, 100, m_hwnd, (HMENU)2, g_hinst, 0);
  }
  return 0;
}
```

We define a global variable to hold the window that is the hot potato that will be passed from one root window to another. When we create a root window, we give it a button called "Reset", and the first root window also gets the hot potato.

```
LRESULT RootWindow::HandleMessage(
                          UINT uMsg, WPARAM wParam, LPARAM lParam)
{
 switch (uMsg) {
 ...
  case WM_COMMAND:
   switch (GET_WM_COMMAND_ID(wParam, lParam)) {
   case 1:
    switch (GET_WM_COMMAND_CMD(wParam, lParam)) {
    case BN_CLICKED:
     SendMessage(m_hwnd, WM_CHANGEUISTATE,
      MAKEWPARAM(UIS_SET, UISF_HIDEACCEL | UISF_HIDEFOCUS), 0);
     break;
    }
    break;
   }
   break;

  case WM_LBUTTONDOWN:
   if (GetParent(g_hwndPotato) != m_hwnd) {
    SetParent(g_hwndPotato, m_hwnd);
   }
   break;

  // delete WM_SIZE handler
  // case WM_SIZE:
  //  if (m_hwndChild) {
  //   SetWindowPos(m_hwndChild, NULL, 0, 0,
  //                GET_X_LPARAM(lParam), GET_Y_LPARAM(lParam),
  //                SWP_NOZORDER | SWP_NOACTIVATE);
  //  }
  //  return 0;

  ...
 }
 return __super::HandleMessage(uMsg, wParam, lParam);
}
```

If the user clicks the Reset button, we hide keyboard accelerators and the focus rectangle. This lets us tinker with the UI state interactively: Click the button to hide accelerators and tap the `Alt` key to bring them back.

If we click in an empty space in the window, then we try to steal the potato window (assuming it's not already ours).

```
int PASCAL
WinMain(HINSTANCE hinst, HINSTANCE, LPSTR, int nShowCmd)
{
 g_hinst = hinst;

 if (SUCCEEDED(CoInitialize(NULL))) {
  InitCommonControls();

  RootWindow *prw1 = RootWindow::Create();
  RootWindow *prw2 = RootWindow::Create();
  if (prw1 && prw2) {
   ShowWindow(prw1->GetHWND(), nShowCmd);
   ShowWindow(prw2->GetHWND(), nShowCmd);
   MSG msg;
   while (GetMessage(&msg, NULL, 0, 0)) {
    if (!IsDialogMessage(prw1->GetHWND(), &msg) &&
        !IsDialogMessage(prw2->GetHWND(), &msg)) {
     TranslateMessage(&msg);
     DispatchMessage(&msg);
    }
   }
  }
  CoUninitialize();
 }
 return 0;
}
```

Finally, our main program creates two top-level windows and lets you play with them.

Observe that we do nothing to synchronize the UI states of the parent window with the potato.

Run this program and click Reset in the window that doesn't have the Potato button. This hides the keyboard accelerator indicator. Now click in the client area of the window without the Potato button. The Potato window moves to that window, and observe that the Potato window still has its keyboard accelerator even though the Reset button doesn't.

Okay, now things get weirder: hit the `Tab` key to put focus on the Potato window. Now hit the `Alt` key to call up keyboard accelerator indicators. Notice that the indicators do not show up on the Reset button. That's because the indicators are already enabled in the Potato window, so it doesn't bother forwarding the message up to the root, since it figures the root would just ignore it anyway.

Click the Reset button to remove keyboard indicators from both the Reset and Potato windows. Go to the other window (where Reset has the keyboard indicators) and click in the client area to steal the potato. Now you have the weird reverse state where the Reset button has a keyboard indicator but the Potato button doesn't. Use the `Tab` to put focus on the

Reset button, and then tap the `Alt` to call up keyboard indicators. Oops, nothing happens because the Reset button already has keyboard indicators, so it decides that there is no work to do.

Okay, so we see that if you have a window tree with a mix of UI states, things act weird. Next time, we'll try to fix it.

Raymond Chen

Follow