How do I prevent users from using the mouse to drag the trackbar thumb to positions that aren't multiples of five? Part 2: Nudging the thumb position

devblogs.microsoft.com/oldnewthing/20181026-00

October 26, 2018

Raymond Chen

Last time, we investigated one approach to the problem of keeping a trackbar position on multiples of five: Just divide everything by five! (And then multiply by five when you read the value.) This works great, except that accessibility tools report the range as 0–20 in one-unit increments, even though you are presenting it to the user as a range of 0–100 in five-unit increments.

So it's back to the drawing board. This time, we'll leave the range at 0–100 and manipulate the trackbar position in the `TRBN_ THUMBPOSCHANGING` notification.

Take our program from last time and make the following changes:

```
#pragma comment(linker, \
    "\"/manifestdependency:type='win32' \
    name='Microsoft.Windows.Common-Controls' \
    version='6.0.0.0' \
    processorArchitecture='*' \
    publicKeyToken='6595b64144ccf1df' \
    language='*'\"")
```

This `#pragma` is a quick way to enable version 6 of the common controls.

```
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
 g_hwndChild = CreateWindow(TRACKBAR_CLASS, TEXT(""),
    WS_CHILD | WS_VISIBLE | TBS_NOTIFYBEFOREMOVE,
   0, 0, 100, 100,
    hwnd, (HMENU)100, g_hinst, 0);

  SendMessage(g_hwndChild, TBM_SETLINESIZE, 0, 5);
  SendMessage(g_hwndChild, TBM_SETPAGESIZE, 0, 20);

  return TRUE;
}
```

The `TBS_NOTIFYBEFOREMOVE` style enables the `TRBN_THUMBPOSCHANGING` notification, which we will take advantage of below. We leave the range at its default of 0–100 and set the line size and page size to the desired multiples of five.

```
LRESULT OnNotify(HWND hwnd, int idCtl, NMHDR* pnm)
{
 if (pnm->hwndFrom == g_hwndChild &&
     pnm->code == TRBN_THUMBPOSCHANGING) {
  auto ptpc = (NMTRBTHUMBPOSCHANGING*)pnm;
  auto pos = ptpc->dwPos;
  auto newpos = (pos + 2) / 5 * 5;
  if (pos != newpos) {
    SendMessage(pnm->hwndFrom, TBM_SETPOS, TRUE, newpos);
    return TRUE; // we moved the thumb, so the control doesn't have to
  }
 }
 return 0;
}
    HANDLE_MSG(hwnd, WM_NOTIFY, OnNotify);
```

When we are being notified that the thumb is about to move, we take the proposed new position and round it to the nearest multiple of five. If that produces a value different from what the control would have done, then we manually set the thumb position to the desired multiple of five and return `TRUE` to tell the trackbar that it shouldn't move the trackbar thumb (because we moved it).

(And again, don't forget that if this is happening in a dialog box, you need to use `DWLP_MSGRESULT` to make the dialog box return a nonzero value from its window procedure.)

Raymond Chen

Follow