Creating a listview with checkboxes on some items but not others

devblogs.microsoft.com/oldnewthing/20140113-00

January 13, 2014

Raymond Chen

Today's Little Program creates a listview with checkboxes on some items but not other.

The `LVS_EX_CHECKBOXES` extended style is really just a convenience style. Everything it does you could have done yourself, with a bit more typing.

- It creates a state image list consisting of an unchecked box (state 1) and a checked box (state 2). You could have done this yourself with `ImageList_Create` followed by a few calls to `DrawFrameControl`.
- When you hit the space bar or click on the check box, the state image toggles between 1 and 2. You could have done this yourself by responding to `LVN_KEYDOWN` (for the space bar), and the mouse notification messages for the clicks. (For the mouse notifications, see if the click was on `LVHT_ONITEMSTATEICON`.)

But still, it's convenient having the listview control do this grunt work for you. But what if you want to remove the check box from some items?

The listview control turns on the state image and toggles it by doing the moral equivalent of a `ListView_SetCheckState` on the item, so all you have to do is respond to the `LVN_ITEM-CHANGING` that comes with any item change and reject the state change.

Start with our scratch program and make these changes. Remember, Little Programs do little or no error checking.

```
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
  g_hwndChild = CreateWindow(WC_LISTVIEW, NULL,
        WS_CHILD | WS_VISIBLE | LVS_REPORT,
        0, 0, 0, 0, hwnd, (HMENU)1, g_hinst, 0);
  ListView_SetExtendedListViewStyle(g_hwndChild,
                                    LVS_EX_CHECKBOXES);
  LVCOLUMN col;
  col.mask = LVCF_TEXT | LVCF_WIDTH;
  col.cx = 200;
  col.pszText = TEXT("Name");
  ListView_InsertColumn(g_hwndChild, 0, &col);
  LVITEM item;
  item.mask = LVIF_TEXT;
  item.iSubItem = 0;
  item.pszText = TEXT("Alpha");
  ListView_InsertItem(g_hwndChild, &item);
  item.pszText = TEXT("Beta");
  ListView_InsertItem(g_hwndChild, &item);
  item.pszText = TEXT("Gamma");
  ListView_InsertItem(g_hwndChild, &item);
  item.pszText = TEXT("Delta");
  ListView_InsertItem(g_hwndChild, &item);
  return TRUE;
}
```

Okay, so far the program adds four items, each with a check box. But let's say we want to remove the check boxes from the even-numbered items.

```
LRESULT
OnNotify(HWND hwnd, int idFrom, NMHDR *pnm)
{
  if (idFrom == 1) {
    switch (pnm->code) {
    case LVN_ITEMCHANGING:
      {
        LPNMLISTVIEW pnmlv = CONTAINING_RECORD(pnm, NMLISTVIEW, hdr);
        if (pnmlv->iItem >= 0 &&
        if (pnmlv->iItem % 2 == 0 &&
            (pnmlv->uChanged & LVIF_STATE)) {
         return TRUE; // reject changes to even-numbered items
        }
      }
      break;
    }
  }
  return 0;
}
    HANDLE_MSG(hwnd, WM_NOTIFY, OnNotify);
```

We add a handler for `LVN_ITEMCHANGING` that says, “If this is a notification for an even-numbered item, and they want to change the state, then block the state change.” This ensures that nobody can turn on the state image, which means that the checkbox never shows up.

Raymond Chen

Follow