What states are possible in a DRAWITEMSTRUCT structure?

devblogs.microsoft.com/oldnewthing/20141211-00

December 11, 2014

Raymond Chen

The `DRAWITEMSTRUCT` structure has an `itemState` member which contains a number of bits describing the state of the item being drawn. How do those states map to the underlying control? Most of the states are rather obvious. For a list box item to be *selected*, it means that the item is part of the selection. But what does *selected* mean for a button? Since people like tables, I'll put the answer in a table:

	Menu	**Listbox**	**Combobox**	**Button**
`CtlType`	`ODT_MENU`	`ODT_LISTBOX`	`ODT_COMBOBOX`	`ODT_BUTTON`
`itemID`	menu item ID	item index or −1	item index or −1	
`ODS_SELECTED`	Selected	Selected	Selected	Pushed
`ODS_GRAYED`	Grayed			
`ODS_DISABLED`	Disabled	Disabled	Disabled	Disabled
`ODS_CHECKED`	Checked			
`ODS_FOCUS`		Focus	Focus	Focus
`ODS_DEFAULT`	Default menu item			
`ODS_HOTLIGHT`	Hover			
`ODS_INACTIVE`	Inactive			
`ODS_NOACCEL`	HideAccel	HideAccel	HideAccel	HideAccel
`ODS_NOFOCUSRECT`		HideFocus	HideFocus	HideFocus
`ODS_COMBOBOXEDIT`			Is edit control	

	Static	Header	Tab	Listview	Sta
`CtlType`	`ODT_STATIC`	`ODT_HEADER`	`ODT_TAB`	`ODT_LISTVIEW`	
`itemID`		item index	item index	item index	par ind
`ODS_SELECTED`		Pushed	Selected	Selected	
`ODS_GRAYED`					
`ODS_DISABLED`	Oops				
`ODS_CHECKED`				AutoChecked	
`ODS_FOCUS`				Focus	
`ODS_DEFAULT`					
`ODS_HOTLIGHT`				Hover	
`ODS_INACTIVE`					
`ODS_NOACCEL`	HideAccel				
`ODS_NOFOCUSRECT`					
`ODS_COMBOBOXEDIT`					

Okay, now that it's all in a table, how do I read the table? A box is blank if the corresponding flag is not currently used by the control type. (No guarantees about the future.) For example, as of this writing, button controls do not set an `itemID`, nor do they ever ask for `ODS_GRAYED`. You may have noticed that the box for `CtlType` is blank for status controls. That's an oops. The status bar control forgot to set the `CtlType` when it sends the `WM_DRAWITEM` message, so the value is uninitialized garbage. The way to detect a status bar control is to check the window handle. (This works in general. You can always detect a control by checking the window handle.) For list boxes and combo boxes, the `itemID` can have the special value `-1` to mean "I am drawing a list box/combo box where no item is selected." For list boxes, this happens when the list box is empty. For combo boxes, this happens when the user types text into the edit box that does not match any of the items in the list portion of the combo box. Most of the other box entries are self-explanatory. For the most part, the flag name matches the conditions under which the corresponding flag is set. For example, the `ODS_FOCUS` flag is set when the list box item being drawn is the selected item. Note that the `ODS_SELECTED` flag is used for button and header controls to indicate that the control should be drawn in the pushed state. For example, the user may have put focus on a button control and pressed the space bar and not yet released it, or the application may have manually set the `BST_PUSHED` state. Header controls can get into a *pushed* state if you enable the `HDS_BUTTONS` style. List view controls set the `ODS_CHECKED` flag if a check box should be drawn over the item. This happens if the `LVS_EX_AUTOCHECKSELECT` extended style is specified and the item is selected. (Normally, the check box is drawn to the side as a state image.) The `ODS_COMBOBOXEDIT` flag is used only by combo box controls. It is set if the item being drawn is the edit portion of a combo box control. If not set, then the item being drawn is in the list box portion of the combo box control. Finally, there is a box marked Oops. The static control is supposed to set `ODS_DISABLED` if the static control is disabled. And that's what happens if you are using the classic static control. However, there is a typo in the the fancy themed static control, and it sets the `ODS_DISBALED` flag incorrectly. If you are owner-drawing a themed static control, and you want to draw differently depending on whether the control is disabled, then you should ignore the `ODS_DISABLED` flag and instead draw the disabled state based on the result of calling `IsWindowEnabled` function.

The bug in the themed static control cannot be fixed for compatibility reasons. I can pretty much guarantee that there is some application which doesn't draw correctly if the `ODS_DISABLED` flag is not set.

Raymond Chen

Follow