
NSMenuItemActionResponder

(informal protocol)

Category Of: NSObject

Declared In: AppKit/NSMenu.h

Protocol Description

This informal protocol allows your application to update the enabled or disabled status of an `NSMenuItem`. It declares only one method, **validateMenuItem:**. By default, every time a user event occurs, `NSMenu` automatically enables and disables each visible menu item based on criteria described in “Automatic Updating of `NSMenuItems`,” below. Implement **validateMenuItem:** in cases where you want to override `NSMenu`’s default enabling scheme.

`NSMenuItems` can be enabled or disabled in two ways: explicitly, by sending the **setEnabled:** message, or automatically, as described below. Automatic updating can be turned on and off with `NSMenu`’s **setAutoenablesItems:** message.

Automatic Updating of `NSMenuItems`

Whenever a user event occurs, the `NSMenu` object updates the status of every one of its visible menu items. To update the status of a menu item, an `NSMenu` tries to find the object that responds to the `NSMenuItem`’s action message. It searches the following objects in order until it finds one that responds to the action message (note that it doesn’t actually send the action message):

- The `NSMenuItem`’s target. If the target is non-**nil**, the search ends here whether the target responds or not.
- The key window’s responder chain, starting with its first responder.
- The key window itself.
- The key window’s delegate.
- The main window’s responder chain, starting with its first responder.
- The main window itself.
- The main window’s delegate.
- The `NSApplication` object.
- The `NSApplication` object’s delegate.

If none of these objects responds to the action message, the menu item is disabled. If `NSMenu` finds an object that responds to the action message, it then checks to see if that object responds to the **validateMenuItem:** method (the method declared in this informal protocol). If **validateMenuItem:** is not

implemented in that object, the menu item is enabled. If it is implemented, the return value of **`validateMenuItem:`** indicates whether the menu item should be enabled or disabled.

Here is an example of using **`validateMenuItem:`** to override automatic enabling. If your application has a Copy menu item that sends the **`copy:`** action message to the first responder, that menu item is automatically enabled any time an object that responds to **`copy:`**, such as an `NSString` object, is the first responder of the key or main window. If you create a class whose instances might become the first responder, and which doesn't support copying of everything it allows the user to select, you should implement

`validateMenuItem:` in that class. **`validateMenuItem:`** will then return `NO` if items that can't be copied are selected (or if no items are selected) and `YES` if all items in the selection can be copied. By implementing **`validateMenuItem:`**, you can have the Copy menu item disabled even though the target object does implement the **`copy:`** method. If your class never permits copying, then you simply omit an implementation of **`copy:`** in that class, and the Copy menu item is disabled automatically whenever an instance of that class is the first responder.

If you send a **`setEnabled:`** message to enable or disable a menu item when automatic updating is turned on (with `NSMenu`'s **`setAutoEnablesItems:`**), other objects might undo what you have done after another user event occurs. Hence you can never be sure that the menu item will remain the way you set it. If your application must use **`setEnabled:`**, turn off the automatic enabling of menu items in order to get predictable results.

Instance Methods

`validateMenuItem:`

– (BOOL)**`validateMenuItem:`**(`NSMenuItem *`)*aMenuItem*

Implemented to override the default action of enabling or disabling *aMenuItem*. The object implementing this method must be the target of *aMenuItem*. It returns `YES` to enable the *aMenuItem*, `NO` to disable it. You can determine which menu item *aMenuItem* is by querying it for its title, tag, or action.

The following example beeps and disables the menu item “Next Record” if the selected line in a table view is the last one; conversely, it beeps and disables the menu item “Prior Record” if the selected row is the first one in the table view. (**`countryKeys`** is an array of names appearing in the table view.)

```
- (BOOL)validateMenuItem:(NSMenuItem *)anItem
{
    int row = [tableView selectedRow];
    if ([[anItem title] isEqualToString:@"Next Record"] &&
        (row == [countryKeys indexOfObject:[countryKeys lastObject]])) {
        NSBeep();
        return NO;
    }
    if ([[anItem title] isEqualToString:@"Prior Record"] && row == 0 ) {
        NSBeep();
        return NO;
    }
}
```

```
    }  
    return YES;  
}
```