| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/MenuSelectionManager.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/swing/MenuElement.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/MutableComboBoxModel.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/MenuSelectionManager.html)    [**NO FRAMES**](http://docs.google.com/MenuSelectionManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

## **javax.swing**

Class MenuSelectionManager

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.swing.MenuSelectionManager**

public class **MenuSelectionManager**extends [Object](http://docs.google.com/java/lang/Object.html)

A MenuSelectionManager owns the selection in menu hierarchy.

| **Field Summary** | |
| --- | --- |
| protected  [ChangeEvent](http://docs.google.com/javax/swing/event/ChangeEvent.html) | [**changeEvent**](http://docs.google.com/javax/swing/MenuSelectionManager.html#changeEvent)            Only one ChangeEvent is needed per button model instance since the event's only state is the source property. |
| protected  [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html) | [**listenerList**](http://docs.google.com/javax/swing/MenuSelectionManager.html#listenerList) |

| **Constructor Summary** | |
| --- | --- |
| [**MenuSelectionManager**](http://docs.google.com/javax/swing/MenuSelectionManager.html#MenuSelectionManager())() |

| **Method Summary** | |
| --- | --- |
| void | [**addChangeListener**](http://docs.google.com/javax/swing/MenuSelectionManager.html#addChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)            Adds a ChangeListener to the button. |
| void | [**clearSelectedPath**](http://docs.google.com/javax/swing/MenuSelectionManager.html#clearSelectedPath())()            Tell the menu selection to close and unselect all the menu components. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**componentForPoint**](http://docs.google.com/javax/swing/MenuSelectionManager.html#componentForPoint(java.awt.Component,%20java.awt.Point))([Component](http://docs.google.com/java/awt/Component.html) source, [Point](http://docs.google.com/java/awt/Point.html) sourcePoint)            Returns the component in the currently selected path which contains sourcePoint. |
| static [MenuSelectionManager](http://docs.google.com/javax/swing/MenuSelectionManager.html) | [**defaultManager**](http://docs.google.com/javax/swing/MenuSelectionManager.html#defaultManager())()            Returns the default menu selection manager. |
| protected  void | [**fireStateChanged**](http://docs.google.com/javax/swing/MenuSelectionManager.html#fireStateChanged())()            Notifies all listeners that have registered interest for notification on this event type. |
| [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html)[] | [**getChangeListeners**](http://docs.google.com/javax/swing/MenuSelectionManager.html#getChangeListeners())()            Returns an array of all the ChangeListeners added to this MenuSelectionManager with addChangeListener(). |
| [MenuElement](http://docs.google.com/javax/swing/MenuElement.html)[] | [**getSelectedPath**](http://docs.google.com/javax/swing/MenuSelectionManager.html#getSelectedPath())()            Returns the path to the currently selected menu item |
| boolean | [**isComponentPartOfCurrentMenu**](http://docs.google.com/javax/swing/MenuSelectionManager.html#isComponentPartOfCurrentMenu(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) c)            Return true if c is part of the currently used menu |
| void | [**processKeyEvent**](http://docs.google.com/javax/swing/MenuSelectionManager.html#processKeyEvent(java.awt.event.KeyEvent))([KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)            When a MenuElement receives an event from a KeyListener, it should never process the event directly. |
| void | [**processMouseEvent**](http://docs.google.com/javax/swing/MenuSelectionManager.html#processMouseEvent(java.awt.event.MouseEvent))([MouseEvent](http://docs.google.com/java/awt/event/MouseEvent.html) event)            When a MenuElement receives an event from a MouseListener, it should never process the event directly. |
| void | [**removeChangeListener**](http://docs.google.com/javax/swing/MenuSelectionManager.html#removeChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)            Removes a ChangeListener from the button. |
| void | [**setSelectedPath**](http://docs.google.com/javax/swing/MenuSelectionManager.html#setSelectedPath(javax.swing.MenuElement%5B%5D))([MenuElement](http://docs.google.com/javax/swing/MenuElement.html)[] path)            Changes the selection in the menu hierarchy. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Field Detail** |
| --- |

### changeEvent

protected transient [ChangeEvent](http://docs.google.com/javax/swing/event/ChangeEvent.html) **changeEvent**

Only one ChangeEvent is needed per button model instance since the event's only state is the source property. The source of events generated is always "this".

### listenerList

protected [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html) **listenerList**

| **Constructor Detail** |
| --- |

### MenuSelectionManager

public **MenuSelectionManager**()

| **Method Detail** |
| --- |

### defaultManager

public static [MenuSelectionManager](http://docs.google.com/javax/swing/MenuSelectionManager.html) **defaultManager**()

Returns the default menu selection manager.

**Returns:**a MenuSelectionManager object

### setSelectedPath

public void **setSelectedPath**([MenuElement](http://docs.google.com/javax/swing/MenuElement.html)[] path)

Changes the selection in the menu hierarchy. The elements in the array are sorted in order from the root menu element to the currently selected menu element.

Note that this method is public but is used by the look and feel engine and should not be called by client applications.

**Parameters:**path - an array of MenuElement objects specifying the selected path

### getSelectedPath

public [MenuElement](http://docs.google.com/javax/swing/MenuElement.html)[] **getSelectedPath**()

Returns the path to the currently selected menu item

**Returns:**an array of MenuElement objects representing the selected path

### clearSelectedPath

public void **clearSelectedPath**()

Tell the menu selection to close and unselect all the menu components. Call this method when a choice has been made

### addChangeListener

public void **addChangeListener**([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)

Adds a ChangeListener to the button.

**Parameters:**l - the listener to add

### removeChangeListener

public void **removeChangeListener**([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)

Removes a ChangeListener from the button.

**Parameters:**l - the listener to remove

### getChangeListeners

public [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html)[] **getChangeListeners**()

Returns an array of all the ChangeListeners added to this MenuSelectionManager with addChangeListener().

**Returns:**all of the ChangeListeners added or an empty array if no listeners have been added**Since:** 1.4

### fireStateChanged

protected void **fireStateChanged**()

Notifies all listeners that have registered interest for notification on this event type. The event instance is created lazily.

**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### processMouseEvent

public void **processMouseEvent**([MouseEvent](http://docs.google.com/java/awt/event/MouseEvent.html) event)

When a MenuElement receives an event from a MouseListener, it should never process the event directly. Instead all MenuElements should call this method with the event.

**Parameters:**event - a MouseEvent object

### componentForPoint

public [Component](http://docs.google.com/java/awt/Component.html) **componentForPoint**([Component](http://docs.google.com/java/awt/Component.html) source,  
 [Point](http://docs.google.com/java/awt/Point.html) sourcePoint)

Returns the component in the currently selected path which contains sourcePoint.

**Parameters:**source - The component in whose coordinate space sourcePoint is givensourcePoint - The point which is being tested **Returns:**The component in the currently selected path which contains sourcePoint (relative to the source component's coordinate space. If sourcePoint is not inside a component on the currently selected path, null is returned.

### processKeyEvent

public void **processKeyEvent**([KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)

When a MenuElement receives an event from a KeyListener, it should never process the event directly. Instead all MenuElements should call this method with the event.

**Parameters:**e - a KeyEvent object

### isComponentPartOfCurrentMenu

public boolean **isComponentPartOfCurrentMenu**([Component](http://docs.google.com/java/awt/Component.html) c)

Return true if c is part of the currently used menu

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/MenuSelectionManager.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/swing/MenuElement.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/MutableComboBoxModel.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/MenuSelectionManager.html)    [**NO FRAMES**](http://docs.google.com/MenuSelectionManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).