| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyboardFocusManager.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/java/awt/JobAttributes.SidesType.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/KeyEventDispatcher.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/KeyboardFocusManager.html)    [**NO FRAMES**](http://docs.google.com/KeyboardFocusManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#35nkun2) |

## **java.awt**

Class KeyboardFocusManager

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.awt.KeyboardFocusManager**

**All Implemented Interfaces:** [KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html), [KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) **Direct Known Subclasses:** [DefaultKeyboardFocusManager](http://docs.google.com/java/awt/DefaultKeyboardFocusManager.html)

public abstract class **KeyboardFocusManager**extends [Object](http://docs.google.com/java/lang/Object.html)implements [KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html), [KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html)

The KeyboardFocusManager is responsible for managing the active and focused Windows, and the current focus owner. The focus owner is defined as the Component in an application that will typically receive all KeyEvents generated by the user. The focused Window is the Window that is, or contains, the focus owner. Only a Frame or a Dialog can be the active Window. The native windowing system may denote the active Window or its children with special decorations, such as a highlighted title bar. The active Window is always either the focused Window, or the first Frame or Dialog that is an owner of the focused Window.

The KeyboardFocusManager is both a centralized location for client code to query for the focus owner and initiate focus changes, and an event dispatcher for all FocusEvents, WindowEvents related to focus, and KeyEvents.

Some browsers partition applets in different code bases into separate contexts, and establish walls between these contexts. In such a scenario, there will be one KeyboardFocusManager per context. Other browsers place all applets into the same context, implying that there will be only a single, global KeyboardFocusManager for all applets. This behavior is implementation-dependent. Consult your browser's documentation for more information. No matter how many contexts there may be, however, there can never be more than one focus owner, focused Window, or active Window, per ClassLoader.

Please see  [How to Use the Focus Subsystem](http://java.sun.com/docs/books/tutorial/uiswing/misc/focus.html), a section in *The Java Tutorial*, and the [Focus Specification](http://docs.google.com/java/awt/doc-files/FocusSpec.html) for more information.

**Since:** 1.4 **See Also:**[Window](http://docs.google.com/java/awt/Window.html), [Frame](http://docs.google.com/java/awt/Frame.html), [Dialog](http://docs.google.com/java/awt/Dialog.html), [FocusEvent](http://docs.google.com/java/awt/event/FocusEvent.html), [WindowEvent](http://docs.google.com/java/awt/event/WindowEvent.html), [KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html)

| **Field Summary** | |
| --- | --- |
| static int | [**BACKWARD\_TRAVERSAL\_KEYS**](http://docs.google.com/java/awt/KeyboardFocusManager.html#BACKWARD_TRAVERSAL_KEYS)            The identifier for the Backward focus traversal keys. |
| static int | [**DOWN\_CYCLE\_TRAVERSAL\_KEYS**](http://docs.google.com/java/awt/KeyboardFocusManager.html#DOWN_CYCLE_TRAVERSAL_KEYS)            The identifier for the Down Cycle focus traversal keys. |
| static int | [**FORWARD\_TRAVERSAL\_KEYS**](http://docs.google.com/java/awt/KeyboardFocusManager.html#FORWARD_TRAVERSAL_KEYS)            The identifier for the Forward focus traversal keys. |
| static int | [**UP\_CYCLE\_TRAVERSAL\_KEYS**](http://docs.google.com/java/awt/KeyboardFocusManager.html#UP_CYCLE_TRAVERSAL_KEYS)            The identifier for the Up Cycle focus traversal keys. |

| **Constructor Summary** | |
| --- | --- |
| [**KeyboardFocusManager**](http://docs.google.com/java/awt/KeyboardFocusManager.html#KeyboardFocusManager())()            Initializes a KeyboardFocusManager. |

| **Method Summary** | |
| --- | --- |
| void | [**addKeyEventDispatcher**](http://docs.google.com/java/awt/KeyboardFocusManager.html#addKeyEventDispatcher(java.awt.KeyEventDispatcher))([KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) dispatcher)            Adds a KeyEventDispatcher to this KeyboardFocusManager's dispatcher chain. |
| void | [**addKeyEventPostProcessor**](http://docs.google.com/java/awt/KeyboardFocusManager.html#addKeyEventPostProcessor(java.awt.KeyEventPostProcessor))([KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) processor)            Adds a KeyEventPostProcessor to this KeyboardFocusManager's post- processor chain. |
| void | [**addPropertyChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.beans.PropertyChangeListener))([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)            Adds a PropertyChangeListener to the listener list. |
| void | [**addPropertyChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))([String](http://docs.google.com/java/lang/String.html) propertyName, [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)            Adds a PropertyChangeListener to the listener list for a specific property. |
| void | [**addVetoableChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.lang.String,%20java.beans.VetoableChangeListener))([String](http://docs.google.com/java/lang/String.html) propertyName, [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)            Adds a VetoableChangeListener to the listener list for a specific property. |
| void | [**addVetoableChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.beans.VetoableChangeListener))([VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)            Adds a VetoableChangeListener to the listener list. |
| void | [**clearGlobalFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#clearGlobalFocusOwner())()            Clears the global focus owner at both the Java and native levels. |
| protected abstract  void | [**dequeueKeyEvents**](http://docs.google.com/java/awt/KeyboardFocusManager.html#dequeueKeyEvents(long,%20java.awt.Component))(long after, [Component](http://docs.google.com/java/awt/Component.html) untilFocused)            Called by the AWT to notify the KeyboardFocusManager that it should cancel delayed dispatching of KeyEvents. |
| protected abstract  void | [**discardKeyEvents**](http://docs.google.com/java/awt/KeyboardFocusManager.html#discardKeyEvents(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) comp)            Called by the AWT to notify the KeyboardFocusManager that it should cancel delayed dispatching of KeyEvents. |
| abstract  boolean | [**dispatchEvent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#dispatchEvent(java.awt.AWTEvent))([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) e)            This method is called by the AWT event dispatcher requesting that the current KeyboardFocusManager dispatch the specified event on its behalf. |
| abstract  boolean | [**dispatchKeyEvent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#dispatchKeyEvent(java.awt.event.KeyEvent))([KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)            Typically this method will be called by dispatchEvent if no other KeyEventDispatcher in the dispatcher chain dispatched the KeyEvent, or if no other KeyEventDispatchers are registered. |
| void | [**downFocusCycle**](http://docs.google.com/java/awt/KeyboardFocusManager.html#downFocusCycle())()            Moves the focus down one focus traversal cycle from the current focus owner, if and only if the current focus owner is a Container that is a focus cycle root. |
| abstract  void | [**downFocusCycle**](http://docs.google.com/java/awt/KeyboardFocusManager.html#downFocusCycle(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) aContainer)            Moves the focus down one focus traversal cycle. |
| protected abstract  void | [**enqueueKeyEvents**](http://docs.google.com/java/awt/KeyboardFocusManager.html#enqueueKeyEvents(long,%20java.awt.Component))(long after, [Component](http://docs.google.com/java/awt/Component.html) untilFocused)            Called by the AWT to notify the KeyboardFocusManager that it should delay dispatching of KeyEvents until the specified Component becomes the focus owner. |
| protected  void | [**firePropertyChange**](http://docs.google.com/java/awt/KeyboardFocusManager.html#firePropertyChange(java.lang.String,%20java.lang.Object,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) propertyName, [Object](http://docs.google.com/java/lang/Object.html) oldValue, [Object](http://docs.google.com/java/lang/Object.html) newValue)            Fires a PropertyChangeEvent in response to a change in a bound property. |
| protected  void | [**fireVetoableChange**](http://docs.google.com/java/awt/KeyboardFocusManager.html#fireVetoableChange(java.lang.String,%20java.lang.Object,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) propertyName, [Object](http://docs.google.com/java/lang/Object.html) oldValue, [Object](http://docs.google.com/java/lang/Object.html) newValue)            Fires a PropertyChangeEvent in response to a change in a vetoable property. |
| void | [**focusNextComponent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#focusNextComponent())()            Focuses the Component after the current focus owner. |
| abstract  void | [**focusNextComponent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#focusNextComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) aComponent)            Focuses the Component after aComponent, typically based on a FocusTraversalPolicy. |
| void | [**focusPreviousComponent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#focusPreviousComponent())()            Focuses the Component before the current focus owner. |
| abstract  void | [**focusPreviousComponent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#focusPreviousComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) aComponent)            Focuses the Component before aComponent, typically based on a FocusTraversalPolicy. |
| [Window](http://docs.google.com/java/awt/Window.html) | [**getActiveWindow**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getActiveWindow())()            Returns the active Window, if the active Window is in the same context as the calling thread. |
| [Container](http://docs.google.com/java/awt/Container.html) | [**getCurrentFocusCycleRoot**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getCurrentFocusCycleRoot())()            Returns the current focus cycle root, if the current focus cycle root is in the same context as the calling thread. |
| static [KeyboardFocusManager](http://docs.google.com/java/awt/KeyboardFocusManager.html) | [**getCurrentKeyboardFocusManager**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getCurrentKeyboardFocusManager())()            Returns the current KeyboardFocusManager instance for the calling thread's context. |
| [Set](http://docs.google.com/java/util/Set.html)<[AWTKeyStroke](http://docs.google.com/java/awt/AWTKeyStroke.html)> | [**getDefaultFocusTraversalKeys**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalKeys(int))(int id)            Returns a Set of default focus traversal keys for a given traversal operation. |
| [FocusTraversalPolicy](http://docs.google.com/java/awt/FocusTraversalPolicy.html) | [**getDefaultFocusTraversalPolicy**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalPolicy())()            Returns the default FocusTraversalPolicy. |
| [Window](http://docs.google.com/java/awt/Window.html) | [**getFocusedWindow**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getFocusedWindow())()            Returns the focused Window, if the focused Window is in the same context as the calling thread. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getFocusOwner())()            Returns the focus owner, if the focus owner is in the same context as the calling thread. |
| protected  [Window](http://docs.google.com/java/awt/Window.html) | [**getGlobalActiveWindow**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalActiveWindow())()            Returns the active Window, even if the calling thread is in a different context than the active Window. |
| protected  [Container](http://docs.google.com/java/awt/Container.html) | [**getGlobalCurrentFocusCycleRoot**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalCurrentFocusCycleRoot())()            Returns the current focus cycle root, even if the calling thread is in a different context than the current focus cycle root. |
| protected  [Window](http://docs.google.com/java/awt/Window.html) | [**getGlobalFocusedWindow**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalFocusedWindow())()            Returns the focused Window, even if the calling thread is in a different context than the focused Window. |
| protected  [Component](http://docs.google.com/java/awt/Component.html) | [**getGlobalFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalFocusOwner())()            Returns the focus owner, even if the calling thread is in a different context than the focus owner. |
| protected  [Component](http://docs.google.com/java/awt/Component.html) | [**getGlobalPermanentFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalPermanentFocusOwner())()            Returns the permanent focus owner, even if the calling thread is in a different context than the permanent focus owner. |
| protected  [List](http://docs.google.com/java/util/List.html)<[KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html)> | [**getKeyEventDispatchers**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getKeyEventDispatchers())()            Returns this KeyboardFocusManager's KeyEventDispatcher chain as a List. |
| protected  [List](http://docs.google.com/java/util/List.html)<[KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html)> | [**getKeyEventPostProcessors**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getKeyEventPostProcessors())()            Returns this KeyboardFocusManager's KeyEventPostProcessor chain as a List. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getPermanentFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPermanentFocusOwner())()            Returns the permanent focus owner, if the permanent focus owner is in the same context as the calling thread. |
| [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html)[] | [**getPropertyChangeListeners**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners())()            Returns an array of all the property change listeners registered on this keyboard focus manager. |
| [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html)[] | [**getPropertyChangeListeners**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners(java.lang.String))([String](http://docs.google.com/java/lang/String.html) propertyName)            Returns an array of all the PropertyChangeListeners associated with the named property. |
| [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html)[] | [**getVetoableChangeListeners**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners())()            Returns an array of all the vetoable change listeners registered on this keyboard focus manager. |
| [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html)[] | [**getVetoableChangeListeners**](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners(java.lang.String))([String](http://docs.google.com/java/lang/String.html) propertyName)            Returns an array of all the VetoableChangeListeners associated with the named property. |
| abstract  boolean | [**postProcessKeyEvent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#postProcessKeyEvent(java.awt.event.KeyEvent))([KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)            This method will be called by dispatchKeyEvent. |
| abstract  void | [**processKeyEvent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#processKeyEvent(java.awt.Component,%20java.awt.event.KeyEvent))([Component](http://docs.google.com/java/awt/Component.html) focusedComponent, [KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)            This method initiates a focus traversal operation if and only if the KeyEvent represents a focus traversal key for the specified focusedComponent. |
| void | [**redispatchEvent**](http://docs.google.com/java/awt/KeyboardFocusManager.html#redispatchEvent(java.awt.Component,%20java.awt.AWTEvent))([Component](http://docs.google.com/java/awt/Component.html) target, [AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) e)            Redispatches an AWTEvent in such a way that the AWT event dispatcher will not recursively request that the KeyboardFocusManager, or any installed KeyEventDispatchers, dispatch the event again. |
| void | [**removeKeyEventDispatcher**](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeKeyEventDispatcher(java.awt.KeyEventDispatcher))([KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) dispatcher)            Removes a KeyEventDispatcher which was previously added to this KeyboardFocusManager's dispatcher chain. |
| void | [**removeKeyEventPostProcessor**](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeKeyEventPostProcessor(java.awt.KeyEventPostProcessor))([KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) processor)            Removes a previously added KeyEventPostProcessor from this KeyboardFocusManager's post-processor chain. |
| void | [**removePropertyChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.beans.PropertyChangeListener))([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)            Removes a PropertyChangeListener from the listener list. |
| void | [**removePropertyChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))([String](http://docs.google.com/java/lang/String.html) propertyName, [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)            Removes a PropertyChangeListener from the listener list for a specific property. |
| void | [**removeVetoableChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.lang.String,%20java.beans.VetoableChangeListener))([String](http://docs.google.com/java/lang/String.html) propertyName, [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)            Removes a VetoableChangeListener from the listener list for a specific property. |
| void | [**removeVetoableChangeListener**](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.beans.VetoableChangeListener))([VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)            Removes a VetoableChangeListener from the listener list. |
| static void | [**setCurrentKeyboardFocusManager**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setCurrentKeyboardFocusManager(java.awt.KeyboardFocusManager))([KeyboardFocusManager](http://docs.google.com/java/awt/KeyboardFocusManager.html) newManager)            Sets the current KeyboardFocusManager instance for the calling thread's context. |
| void | [**setDefaultFocusTraversalKeys**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalKeys(int,%20java.util.Set))(int id, [Set](http://docs.google.com/java/util/Set.html)<? extends [AWTKeyStroke](http://docs.google.com/java/awt/AWTKeyStroke.html)> keystrokes)            Sets the default focus traversal keys for a given traversal operation. |
| void | [**setDefaultFocusTraversalPolicy**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalPolicy(java.awt.FocusTraversalPolicy))([FocusTraversalPolicy](http://docs.google.com/java/awt/FocusTraversalPolicy.html) defaultPolicy)            Sets the default FocusTraversalPolicy. |
| protected  void | [**setGlobalActiveWindow**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalActiveWindow(java.awt.Window))([Window](http://docs.google.com/java/awt/Window.html) activeWindow)            Sets the active Window. |
| void | [**setGlobalCurrentFocusCycleRoot**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalCurrentFocusCycleRoot(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) newFocusCycleRoot)            Sets the current focus cycle root. |
| protected  void | [**setGlobalFocusedWindow**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalFocusedWindow(java.awt.Window))([Window](http://docs.google.com/java/awt/Window.html) focusedWindow)            Sets the focused Window. |
| protected  void | [**setGlobalFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalFocusOwner(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) focusOwner)            Sets the focus owner. |
| protected  void | [**setGlobalPermanentFocusOwner**](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalPermanentFocusOwner(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) permanentFocusOwner)            Sets the permanent focus owner. |
| void | [**upFocusCycle**](http://docs.google.com/java/awt/KeyboardFocusManager.html#upFocusCycle())()            Moves the focus up one focus traversal cycle from the current focus owner. |
| abstract  void | [**upFocusCycle**](http://docs.google.com/java/awt/KeyboardFocusManager.html#upFocusCycle(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) aComponent)            Moves the focus up one focus traversal cycle. |

| **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** |
| --- |

### FORWARD\_TRAVERSAL\_KEYS

public static final int **FORWARD\_TRAVERSAL\_KEYS**

The identifier for the Forward focus traversal keys.

**See Also:**[setDefaultFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalKeys(int,%20java.util.Set)), [getDefaultFocusTraversalKeys(int)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalKeys(int)), [Component.setFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [Component.getFocusTraversalKeys(int)](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int)), [Constant Field Values](http://docs.google.com/constant-values.html#java.awt.KeyboardFocusManager.FORWARD_TRAVERSAL_KEYS)

### BACKWARD\_TRAVERSAL\_KEYS

public static final int **BACKWARD\_TRAVERSAL\_KEYS**

The identifier for the Backward focus traversal keys.

**See Also:**[setDefaultFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalKeys(int,%20java.util.Set)), [getDefaultFocusTraversalKeys(int)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalKeys(int)), [Component.setFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [Component.getFocusTraversalKeys(int)](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int)), [Constant Field Values](http://docs.google.com/constant-values.html#java.awt.KeyboardFocusManager.BACKWARD_TRAVERSAL_KEYS)

### UP\_CYCLE\_TRAVERSAL\_KEYS

public static final int **UP\_CYCLE\_TRAVERSAL\_KEYS**

The identifier for the Up Cycle focus traversal keys.

**See Also:**[setDefaultFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalKeys(int,%20java.util.Set)), [getDefaultFocusTraversalKeys(int)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalKeys(int)), [Component.setFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [Component.getFocusTraversalKeys(int)](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int)), [Constant Field Values](http://docs.google.com/constant-values.html#java.awt.KeyboardFocusManager.UP_CYCLE_TRAVERSAL_KEYS)

### DOWN\_CYCLE\_TRAVERSAL\_KEYS

public static final int **DOWN\_CYCLE\_TRAVERSAL\_KEYS**

The identifier for the Down Cycle focus traversal keys.

**See Also:**[setDefaultFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalKeys(int,%20java.util.Set)), [getDefaultFocusTraversalKeys(int)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalKeys(int)), [Component.setFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [Component.getFocusTraversalKeys(int)](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int)), [Constant Field Values](http://docs.google.com/constant-values.html#java.awt.KeyboardFocusManager.DOWN_CYCLE_TRAVERSAL_KEYS)

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

### KeyboardFocusManager

public **KeyboardFocusManager**()

Initializes a KeyboardFocusManager.

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

### getCurrentKeyboardFocusManager

public static [KeyboardFocusManager](http://docs.google.com/java/awt/KeyboardFocusManager.html) **getCurrentKeyboardFocusManager**()

Returns the current KeyboardFocusManager instance for the calling thread's context.

**Returns:**this thread's context's KeyboardFocusManager**See Also:**[setCurrentKeyboardFocusManager(java.awt.KeyboardFocusManager)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setCurrentKeyboardFocusManager(java.awt.KeyboardFocusManager))

### setCurrentKeyboardFocusManager

public static void **setCurrentKeyboardFocusManager**([KeyboardFocusManager](http://docs.google.com/java/awt/KeyboardFocusManager.html) newManager)  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Sets the current KeyboardFocusManager instance for the calling thread's context. If null is specified, then the current KeyboardFocusManager is replaced with a new instance of DefaultKeyboardFocusManager.

If a SecurityManager is installed, the calling thread must be granted the AWTPermission "replaceKeyboardFocusManager" in order to replace the the current KeyboardFocusManager. If this permission is not granted, this method will throw a SecurityException, and the current KeyboardFocusManager will be unchanged.

**Parameters:**newManager - the new KeyboardFocusManager for this thread's context **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if the calling thread does not have permission to replace the current KeyboardFocusManager**See Also:**[getCurrentKeyboardFocusManager()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getCurrentKeyboardFocusManager()), [DefaultKeyboardFocusManager](http://docs.google.com/java/awt/DefaultKeyboardFocusManager.html)

### getFocusOwner

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

Returns the focus owner, if the focus owner is in the same context as the calling thread. The focus owner is defined as the Component in an application that will typically receive all KeyEvents generated by the user. KeyEvents which map to the focus owner's focus traversal keys will not be delivered if focus traversal keys are enabled for the focus owner. In addition, KeyEventDispatchers may retarget or consume KeyEvents before they reach the focus owner.

**Returns:**the focus owner, or null if the focus owner is not a member of the calling thread's context**See Also:**[getGlobalFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalFocusOwner()), [setGlobalFocusOwner(java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalFocusOwner(java.awt.Component))

### getGlobalFocusOwner

protected [Component](http://docs.google.com/java/awt/Component.html) **getGlobalFocusOwner**()  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Returns the focus owner, even if the calling thread is in a different context than the focus owner. The focus owner is defined as the Component in an application that will typically receive all KeyEvents generated by the user. KeyEvents which map to the focus owner's focus traversal keys will not be delivered if focus traversal keys are enabled for the focus owner. In addition, KeyEventDispatchers may retarget or consume KeyEvents before they reach the focus owner.

This method will throw a SecurityException if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context.

**Returns:**the focus owner **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context**See Also:**[getFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getFocusOwner()), [setGlobalFocusOwner(java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalFocusOwner(java.awt.Component))

### setGlobalFocusOwner

protected void **setGlobalFocusOwner**([Component](http://docs.google.com/java/awt/Component.html) focusOwner)

Sets the focus owner. The operation will be cancelled if the Component is not focusable. The focus owner is defined as the Component in an application that will typically receive all KeyEvents generated by the user. KeyEvents which map to the focus owner's focus traversal keys will not be delivered if focus traversal keys are enabled for the focus owner. In addition, KeyEventDispatchers may retarget or consume KeyEvents before they reach the focus owner.

This method does not actually set the focus to the specified Component. It merely stores the value to be subsequently returned by getFocusOwner(). Use Component.requestFocus() or Component.requestFocusInWindow() to change the focus owner, subject to platform limitations.

**Parameters:**focusOwner - the focus owner**See Also:**[getFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getFocusOwner()), [getGlobalFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalFocusOwner()), [Component.requestFocus()](http://docs.google.com/java/awt/Component.html#requestFocus()), [Component.requestFocusInWindow()](http://docs.google.com/java/awt/Component.html#requestFocusInWindow()), [Component.isFocusable()](http://docs.google.com/java/awt/Component.html#isFocusable())

### clearGlobalFocusOwner

public void **clearGlobalFocusOwner**()

Clears the global focus owner at both the Java and native levels. If there exists a focus owner, that Component will receive a permanent FOCUS\_LOST event. After this operation completes, the native windowing system will discard all user-generated KeyEvents until the user selects a new Component to receive focus, or a Component is given focus explicitly via a call to requestFocus(). This operation does not change the focused or active Windows.

**See Also:**[Component.requestFocus()](http://docs.google.com/java/awt/Component.html#requestFocus()), [FocusEvent.FOCUS\_LOST](http://docs.google.com/java/awt/event/FocusEvent.html#FOCUS_LOST)

### getPermanentFocusOwner

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

Returns the permanent focus owner, if the permanent focus owner is in the same context as the calling thread. The permanent focus owner is defined as the last Component in an application to receive a permanent FOCUS\_GAINED event. The focus owner and permanent focus owner are equivalent unless a temporary focus change is currently in effect. In such a situation, the permanent focus owner will again be the focus owner when the temporary focus change ends.

**Returns:**the permanent focus owner, or null if the permanent focus owner is not a member of the calling thread's context**See Also:**[getGlobalPermanentFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalPermanentFocusOwner()), [setGlobalPermanentFocusOwner(java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalPermanentFocusOwner(java.awt.Component))

### getGlobalPermanentFocusOwner

protected [Component](http://docs.google.com/java/awt/Component.html) **getGlobalPermanentFocusOwner**()  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Returns the permanent focus owner, even if the calling thread is in a different context than the permanent focus owner. The permanent focus owner is defined as the last Component in an application to receive a permanent FOCUS\_GAINED event. The focus owner and permanent focus owner are equivalent unless a temporary focus change is currently in effect. In such a situation, the permanent focus owner will again be the focus owner when the temporary focus change ends.

This method will throw a SecurityException if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context.

**Returns:**the permanent focus owner **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context**See Also:**[getPermanentFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPermanentFocusOwner()), [setGlobalPermanentFocusOwner(java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalPermanentFocusOwner(java.awt.Component))

### setGlobalPermanentFocusOwner

protected void **setGlobalPermanentFocusOwner**([Component](http://docs.google.com/java/awt/Component.html) permanentFocusOwner)

Sets the permanent focus owner. The operation will be cancelled if the Component is not focusable. The permanent focus owner is defined as the last Component in an application to receive a permanent FOCUS\_GAINED event. The focus owner and permanent focus owner are equivalent unless a temporary focus change is currently in effect. In such a situation, the permanent focus owner will again be the focus owner when the temporary focus change ends.

This method does not actually set the focus to the specified Component. It merely stores the value to be subsequently returned by getPermanentFocusOwner(). Use Component.requestFocus() or Component.requestFocusInWindow() to change the focus owner, subject to platform limitations.

**Parameters:**permanentFocusOwner - the permanent focus owner**See Also:**[getPermanentFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPermanentFocusOwner()), [getGlobalPermanentFocusOwner()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalPermanentFocusOwner()), [Component.requestFocus()](http://docs.google.com/java/awt/Component.html#requestFocus()), [Component.requestFocusInWindow()](http://docs.google.com/java/awt/Component.html#requestFocusInWindow()), [Component.isFocusable()](http://docs.google.com/java/awt/Component.html#isFocusable())

### getFocusedWindow

public [Window](http://docs.google.com/java/awt/Window.html) **getFocusedWindow**()

Returns the focused Window, if the focused Window is in the same context as the calling thread. The focused Window is the Window that is or contains the focus owner.

**Returns:**the focused Window, or null if the focused Window is not a member of the calling thread's context**See Also:**[getGlobalFocusedWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalFocusedWindow()), [setGlobalFocusedWindow(java.awt.Window)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalFocusedWindow(java.awt.Window))

### getGlobalFocusedWindow

protected [Window](http://docs.google.com/java/awt/Window.html) **getGlobalFocusedWindow**()  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Returns the focused Window, even if the calling thread is in a different context than the focused Window. The focused Window is the Window that is or contains the focus owner.

This method will throw a SecurityException if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context.

**Returns:**the focused Window **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context**See Also:**[getFocusedWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getFocusedWindow()), [setGlobalFocusedWindow(java.awt.Window)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalFocusedWindow(java.awt.Window))

### setGlobalFocusedWindow

protected void **setGlobalFocusedWindow**([Window](http://docs.google.com/java/awt/Window.html) focusedWindow)

Sets the focused Window. The focused Window is the Window that is or contains the focus owner. The operation will be cancelled if the specified Window to focus is not a focusable Window.

This method does not actually change the focused Window as far as the native windowing system is concerned. It merely stores the value to be subsequently returned by getFocusedWindow(). Use Component.requestFocus() or Component.requestFocusInWindow() to change the focused Window, subject to platform limitations.

**Parameters:**focusedWindow - the focused Window**See Also:**[getFocusedWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getFocusedWindow()), [getGlobalFocusedWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalFocusedWindow()), [Component.requestFocus()](http://docs.google.com/java/awt/Component.html#requestFocus()), [Component.requestFocusInWindow()](http://docs.google.com/java/awt/Component.html#requestFocusInWindow()), [Window.isFocusableWindow()](http://docs.google.com/java/awt/Window.html#isFocusableWindow())

### getActiveWindow

public [Window](http://docs.google.com/java/awt/Window.html) **getActiveWindow**()

Returns the active Window, if the active Window is in the same context as the calling thread. Only a Frame or a Dialog can be the active Window. The native windowing system may denote the active Window or its children with special decorations, such as a highlighted title bar. The active Window is always either the focused Window, or the first Frame or Dialog that is an owner of the focused Window.

**Returns:**the active Window, or null if the active Window is not a member of the calling thread's context**See Also:**[getGlobalActiveWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalActiveWindow()), [setGlobalActiveWindow(java.awt.Window)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalActiveWindow(java.awt.Window))

### getGlobalActiveWindow

protected [Window](http://docs.google.com/java/awt/Window.html) **getGlobalActiveWindow**()  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Returns the active Window, even if the calling thread is in a different context than the active Window. Only a Frame or a Dialog can be the active Window. The native windowing system may denote the active Window or its children with special decorations, such as a highlighted title bar. The active Window is always either the focused Window, or the first Frame or Dialog that is an owner of the focused Window.

This method will throw a SecurityException if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context.

**Returns:**the active Window **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context**See Also:**[getActiveWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getActiveWindow()), [setGlobalActiveWindow(java.awt.Window)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalActiveWindow(java.awt.Window))

### setGlobalActiveWindow

protected void **setGlobalActiveWindow**([Window](http://docs.google.com/java/awt/Window.html) activeWindow)

Sets the active Window. Only a Frame or a Dialog can be the active Window. The native windowing system may denote the active Window or its children with special decorations, such as a highlighted title bar. The active Window is always either the focused Window, or the first Frame or Dialog that is an owner of the focused Window.

This method does not actually change the active Window as far as the native windowing system is concerned. It merely stores the value to be subsequently returned by getActiveWindow(). Use Component.requestFocus() or Component.requestFocusInWindow()to change the active Window, subject to platform limitations.

**Parameters:**activeWindow - the active Window**See Also:**[getActiveWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getActiveWindow()), [getGlobalActiveWindow()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalActiveWindow()), [Component.requestFocus()](http://docs.google.com/java/awt/Component.html#requestFocus()), [Component.requestFocusInWindow()](http://docs.google.com/java/awt/Component.html#requestFocusInWindow())

### getDefaultFocusTraversalPolicy

public [FocusTraversalPolicy](http://docs.google.com/java/awt/FocusTraversalPolicy.html) **getDefaultFocusTraversalPolicy**()

Returns the default FocusTraversalPolicy. Top-level components use this value on their creation to initialize their own focus traversal policy by explicit call to Container.setFocusTraversalPolicy.

**Returns:**the default FocusTraversalPolicy. null will never be returned.**See Also:**[setDefaultFocusTraversalPolicy(java.awt.FocusTraversalPolicy)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalPolicy(java.awt.FocusTraversalPolicy)), [Container.setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)](http://docs.google.com/java/awt/Container.html#setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)), [Container.getFocusTraversalPolicy()](http://docs.google.com/java/awt/Container.html#getFocusTraversalPolicy())

### setDefaultFocusTraversalPolicy

public void **setDefaultFocusTraversalPolicy**([FocusTraversalPolicy](http://docs.google.com/java/awt/FocusTraversalPolicy.html) defaultPolicy)

Sets the default FocusTraversalPolicy. Top-level components use this value on their creation to initialize their own focus traversal policy by explicit call to Container.setFocusTraversalPolicy. Note: this call doesn't affect already created components as they have their policy initialized. Only new components will use this policy as their default policy.

**Parameters:**defaultPolicy - the new, default FocusTraversalPolicy **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if defaultPolicy is null**See Also:**[getDefaultFocusTraversalPolicy()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalPolicy()), [Container.setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)](http://docs.google.com/java/awt/Container.html#setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)), [Container.getFocusTraversalPolicy()](http://docs.google.com/java/awt/Container.html#getFocusTraversalPolicy())

### setDefaultFocusTraversalKeys

public void **setDefaultFocusTraversalKeys**(int id,  
 [Set](http://docs.google.com/java/util/Set.html)<? extends [AWTKeyStroke](http://docs.google.com/java/awt/AWTKeyStroke.html)> keystrokes)

Sets the default focus traversal keys for a given traversal operation. This traversal key Set will be in effect on all Windows that have no such Set of their own explicitly defined. This Set will also be inherited, recursively, by any child Component of those Windows that has no such Set of its own explicitly defined.

The default values for the default focus traversal keys are implementation-dependent. Sun recommends that all implementations for a particular native platform use the same default values. The recommendations for Windows and Unix are listed below. These recommendations are used in the Sun AWT implementations.

| Identifier | Meaning | Default |
| --- | --- | --- |
| KeyboardFocusManager.FORWARD\_TRAVERSAL\_KEYS | Normal forward keyboard traversal | TAB on KEY\_PRESSED, CTRL-TAB on KEY\_PRESSED |
| KeyboardFocusManager.BACKWARD\_TRAVERSAL\_KEYS | Normal reverse keyboard traversal | SHIFT-TAB on KEY\_PRESSED, CTRL-SHIFT-TAB on KEY\_PRESSED |
| KeyboardFocusManager.UP\_CYCLE\_TRAVERSAL\_KEYS | Go up one focus traversal cycle | none |
| KeyboardFocusManager.DOWN\_CYCLE\_TRAVERSAL\_KEYS | Go down one focus traversal cycle | none |

To disable a traversal key, use an empty Set; Collections.EMPTY\_SET is recommended.

Using the AWTKeyStroke API, client code can specify on which of two specific KeyEvents, KEY\_PRESSED or KEY\_RELEASED, the focus traversal operation will occur. Regardless of which KeyEvent is specified, however, all KeyEvents related to the focus traversal key, including the associated KEY\_TYPED event, will be consumed, and will not be dispatched to any Component. It is a runtime error to specify a KEY\_TYPED event as mapping to a focus traversal operation, or to map the same event to multiple default focus traversal operations.

**Parameters:**id - one of KeyboardFocusManager.FORWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.BACKWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.UP\_CYCLE\_TRAVERSAL\_KEYS, or KeyboardFocusManager.DOWN\_CYCLE\_TRAVERSAL\_KEYSkeystrokes - the Set of AWTKeyStrokes for the specified operation **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if id is not one of KeyboardFocusManager.FORWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.BACKWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.UP\_CYCLE\_TRAVERSAL\_KEYS, or KeyboardFocusManager.DOWN\_CYCLE\_TRAVERSAL\_KEYS, or if keystrokes is null, or if keystrokes contains null, or if any Object in keystrokes is not an AWTKeyStroke, or if any keystroke represents a KEY\_TYPED event, or if any keystroke already maps to another default focus traversal operation**See Also:**[getDefaultFocusTraversalKeys(int)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getDefaultFocusTraversalKeys(int)), [Component.setFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [Component.getFocusTraversalKeys(int)](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int))

### getDefaultFocusTraversalKeys

public [Set](http://docs.google.com/java/util/Set.html)<[AWTKeyStroke](http://docs.google.com/java/awt/AWTKeyStroke.html)> **getDefaultFocusTraversalKeys**(int id)

Returns a Set of default focus traversal keys for a given traversal operation. This traversal key Set will be in effect on all Windows that have no such Set of their own explicitly defined. This Set will also be inherited, recursively, by any child Component of those Windows that has no such Set of its own explicitly defined. (See setDefaultFocusTraversalKeys for a full description of each operation.)

**Parameters:**id - one of KeyboardFocusManager.FORWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.BACKWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.UP\_CYCLE\_TRAVERSAL\_KEYS, or KeyboardFocusManager.DOWN\_CYCLE\_TRAVERSAL\_KEYS **Returns:**the Set of AWTKeyStrokes for the specified operation; the Set will be unmodifiable, and may be empty; null will never be returned **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if id is not one of KeyboardFocusManager.FORWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.BACKWARD\_TRAVERSAL\_KEYS, KeyboardFocusManager.UP\_CYCLE\_TRAVERSAL\_KEYS, or KeyboardFocusManager.DOWN\_CYCLE\_TRAVERSAL\_KEYS**See Also:**[setDefaultFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setDefaultFocusTraversalKeys(int,%20java.util.Set)), [Component.setFocusTraversalKeys(int, java.util.Set)](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [Component.getFocusTraversalKeys(int)](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int))

### getCurrentFocusCycleRoot

public [Container](http://docs.google.com/java/awt/Container.html) **getCurrentFocusCycleRoot**()

Returns the current focus cycle root, if the current focus cycle root is in the same context as the calling thread. If the focus owner is itself a focus cycle root, then it may be ambiguous as to which Components represent the next and previous Components to focus during normal focus traversal. In that case, the current focus cycle root is used to differentiate among the possibilities.

This method is intended to be used only by KeyboardFocusManagers and focus implementations. It is not for general client use.

**Returns:**the current focus cycle root, or null if the current focus cycle root is not a member of the calling thread's context**See Also:**[getGlobalCurrentFocusCycleRoot()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalCurrentFocusCycleRoot()), [setGlobalCurrentFocusCycleRoot(java.awt.Container)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalCurrentFocusCycleRoot(java.awt.Container))

### getGlobalCurrentFocusCycleRoot

protected [Container](http://docs.google.com/java/awt/Container.html) **getGlobalCurrentFocusCycleRoot**()  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Returns the current focus cycle root, even if the calling thread is in a different context than the current focus cycle root. If the focus owner is itself a focus cycle root, then it may be ambiguous as to which Components represent the next and previous Components to focus during normal focus traversal. In that case, the current focus cycle root is used to differentiate among the possibilities.

This method will throw a SecurityException if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context.

**Returns:**the current focus cycle root, or null if the current focus cycle root is not a member of the calling thread's context **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this KeyboardFocusManager is not the current KeyboardFocusManager for the calling thread's context**See Also:**[getCurrentFocusCycleRoot()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getCurrentFocusCycleRoot()), [setGlobalCurrentFocusCycleRoot(java.awt.Container)](http://docs.google.com/java/awt/KeyboardFocusManager.html#setGlobalCurrentFocusCycleRoot(java.awt.Container))

### setGlobalCurrentFocusCycleRoot

public void **setGlobalCurrentFocusCycleRoot**([Container](http://docs.google.com/java/awt/Container.html) newFocusCycleRoot)

Sets the current focus cycle root. If the focus owner is itself a focus cycle root, then it may be ambiguous as to which Components represent the next and previous Components to focus during normal focus traversal. In that case, the current focus cycle root is used to differentiate among the possibilities.

This method is intended to be used only by KeyboardFocusManagers and focus implementations. It is not for general client use.

**Parameters:**newFocusCycleRoot - the new focus cycle root**See Also:**[getCurrentFocusCycleRoot()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getCurrentFocusCycleRoot()), [getGlobalCurrentFocusCycleRoot()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getGlobalCurrentFocusCycleRoot())

### addPropertyChangeListener

public void **addPropertyChangeListener**([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)

Adds a PropertyChangeListener to the listener list. The listener is registered for all bound properties of this class, including the following:

* whether the KeyboardFocusManager is currently managing focus for this application or applet's browser context ("managingFocus")
* the focus owner ("focusOwner")
* the permanent focus owner ("permanentFocusOwner")
* the focused Window ("focusedWindow")
* the active Window ("activeWindow")
* the default focus traversal policy ("defaultFocusTraversalPolicy")
* the Set of default FORWARD\_TRAVERSAL\_KEYS ("forwardDefaultFocusTraversalKeys")
* the Set of default BACKWARD\_TRAVERSAL\_KEYS ("backwardDefaultFocusTraversalKeys")
* the Set of default UP\_CYCLE\_TRAVERSAL\_KEYS ("upCycleDefaultFocusTraversalKeys")
* the Set of default DOWN\_CYCLE\_TRAVERSAL\_KEYS ("downCycleDefaultFocusTraversalKeys")
* the current focus cycle root ("currentFocusCycleRoot")

If listener is null, no exception is thrown and no action is performed.

**Parameters:**listener - the PropertyChangeListener to be added**See Also:**[removePropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.beans.PropertyChangeListener)), [getPropertyChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners()), [addPropertyChangeListener(java.lang.String,java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))

### removePropertyChangeListener

public void **removePropertyChangeListener**([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)

Removes a PropertyChangeListener from the listener list. This method should be used to remove the PropertyChangeListeners that were registered for all bound properties of this class.

If listener is null, no exception is thrown and no action is performed.

**Parameters:**listener - the PropertyChangeListener to be removed**See Also:**[addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.beans.PropertyChangeListener)), [getPropertyChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners()), [removePropertyChangeListener(java.lang.String,java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))

### getPropertyChangeListeners

public [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html)[] **getPropertyChangeListeners**()

Returns an array of all the property change listeners registered on this keyboard focus manager.

**Returns:**all of this keyboard focus manager's PropertyChangeListeners or an empty array if no property change listeners are currently registered**Since:** 1.4 **See Also:**[addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.beans.PropertyChangeListener)), [removePropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.beans.PropertyChangeListener)), [getPropertyChangeListeners(java.lang.String)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners(java.lang.String))

### addPropertyChangeListener

public void **addPropertyChangeListener**([String](http://docs.google.com/java/lang/String.html) propertyName,  
 [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)

Adds a PropertyChangeListener to the listener list for a specific property. The specified property may be user-defined, or one of the following:

* whether the KeyboardFocusManager is currently managing focus for this application or applet's browser context ("managingFocus")
* the focus owner ("focusOwner")
* the permanent focus owner ("permanentFocusOwner")
* the focused Window ("focusedWindow")
* the active Window ("activeWindow")
* the default focus traversal policy ("defaultFocusTraversalPolicy")
* the Set of default FORWARD\_TRAVERSAL\_KEYS ("forwardDefaultFocusTraversalKeys")
* the Set of default BACKWARD\_TRAVERSAL\_KEYS ("backwardDefaultFocusTraversalKeys")
* the Set of default UP\_CYCLE\_TRAVERSAL\_KEYS ("upCycleDefaultFocusTraversalKeys")
* the Set of default DOWN\_CYCLE\_TRAVERSAL\_KEYS ("downCycleDefaultFocusTraversalKeys")
* the current focus cycle root ("currentFocusCycleRoot")

If listener is null, no exception is thrown and no action is performed.

**Parameters:**propertyName - one of the property names listed abovelistener - the PropertyChangeListener to be added**See Also:**[addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.beans.PropertyChangeListener)), [removePropertyChangeListener(java.lang.String,java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [getPropertyChangeListeners(java.lang.String)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners(java.lang.String))

### removePropertyChangeListener

public void **removePropertyChangeListener**([String](http://docs.google.com/java/lang/String.html) propertyName,  
 [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)

Removes a PropertyChangeListener from the listener list for a specific property. This method should be used to remove PropertyChangeListeners that were registered for a specific bound property.

If listener is null, no exception is thrown and no action is performed.

**Parameters:**propertyName - a valid property namelistener - the PropertyChangeListener to be removed**See Also:**[addPropertyChangeListener(java.lang.String,java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [getPropertyChangeListeners(java.lang.String)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getPropertyChangeListeners(java.lang.String)), [removePropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.beans.PropertyChangeListener))

### getPropertyChangeListeners

public [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html)[] **getPropertyChangeListeners**([String](http://docs.google.com/java/lang/String.html) propertyName)

Returns an array of all the PropertyChangeListeners associated with the named property.

**Returns:**all of the PropertyChangeListeners associated with the named property or an empty array if no such listeners have been added.**Since:** 1.4 **See Also:**[addPropertyChangeListener(java.lang.String,java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [removePropertyChangeListener(java.lang.String,java.beans.PropertyChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))

### firePropertyChange

protected void **firePropertyChange**([String](http://docs.google.com/java/lang/String.html) propertyName,  
 [Object](http://docs.google.com/java/lang/Object.html) oldValue,  
 [Object](http://docs.google.com/java/lang/Object.html) newValue)

Fires a PropertyChangeEvent in response to a change in a bound property. The event will be delivered to all registered PropertyChangeListeners. No event will be delivered if oldValue and newValue are the same.

**Parameters:**propertyName - the name of the property that has changedoldValue - the property's previous valuenewValue - the property's new value

### addVetoableChangeListener

public void **addVetoableChangeListener**([VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)

Adds a VetoableChangeListener to the listener list. The listener is registered for all vetoable properties of this class, including the following:

* the focus owner ("focusOwner")
* the permanent focus owner ("permanentFocusOwner")
* the focused Window ("focusedWindow")
* the active Window ("activeWindow")

If listener is null, no exception is thrown and no action is performed.

**Parameters:**listener - the VetoableChangeListener to be added**See Also:**[removeVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.beans.VetoableChangeListener)), [getVetoableChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners()), [addVetoableChangeListener(java.lang.String,java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.lang.String,%20java.beans.VetoableChangeListener))

### removeVetoableChangeListener

public void **removeVetoableChangeListener**([VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)

Removes a VetoableChangeListener from the listener list. This method should be used to remove the VetoableChangeListeners that were registered for all vetoable properties of this class.

If listener is null, no exception is thrown and no action is performed.

**Parameters:**listener - the VetoableChangeListener to be removed**See Also:**[addVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.beans.VetoableChangeListener)), [getVetoableChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners()), [removeVetoableChangeListener(java.lang.String,java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.lang.String,%20java.beans.VetoableChangeListener))

### getVetoableChangeListeners

public [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html)[] **getVetoableChangeListeners**()

Returns an array of all the vetoable change listeners registered on this keyboard focus manager.

**Returns:**all of this keyboard focus manager's VetoableChangeListeners or an empty array if no vetoable change listeners are currently registered**Since:** 1.4 **See Also:**[addVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.beans.VetoableChangeListener)), [removeVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.beans.VetoableChangeListener)), [getVetoableChangeListeners(java.lang.String)](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners(java.lang.String))

### addVetoableChangeListener

public void **addVetoableChangeListener**([String](http://docs.google.com/java/lang/String.html) propertyName,  
 [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)

Adds a VetoableChangeListener to the listener list for a specific property. The specified property may be user-defined, or one of the following:

* the focus owner ("focusOwner")
* the permanent focus owner ("permanentFocusOwner")
* the focused Window ("focusedWindow")
* the active Window ("activeWindow")

If listener is null, no exception is thrown and no action is performed.

**Parameters:**propertyName - one of the property names listed abovelistener - the VetoableChangeListener to be added**See Also:**[addVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.beans.VetoableChangeListener)), [removeVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.beans.VetoableChangeListener)), [getVetoableChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners())

### removeVetoableChangeListener

public void **removeVetoableChangeListener**([String](http://docs.google.com/java/lang/String.html) propertyName,  
 [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html) listener)

Removes a VetoableChangeListener from the listener list for a specific property. This method should be used to remove VetoableChangeListeners that were registered for a specific bound property.

If listener is null, no exception is thrown and no action is performed.

**Parameters:**propertyName - a valid property namelistener - the VetoableChangeListener to be removed**See Also:**[addVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.beans.VetoableChangeListener)), [getVetoableChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners()), [removeVetoableChangeListener(java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.beans.VetoableChangeListener))

### getVetoableChangeListeners

public [VetoableChangeListener](http://docs.google.com/java/beans/VetoableChangeListener.html)[] **getVetoableChangeListeners**([String](http://docs.google.com/java/lang/String.html) propertyName)

Returns an array of all the VetoableChangeListeners associated with the named property.

**Returns:**all of the VetoableChangeListeners associated with the named property or an empty array if no such listeners have been added.**Since:** 1.4 **See Also:**[addVetoableChangeListener(java.lang.String,java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addVetoableChangeListener(java.lang.String,%20java.beans.VetoableChangeListener)), [removeVetoableChangeListener(java.lang.String,java.beans.VetoableChangeListener)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeVetoableChangeListener(java.lang.String,%20java.beans.VetoableChangeListener)), [getVetoableChangeListeners()](http://docs.google.com/java/awt/KeyboardFocusManager.html#getVetoableChangeListeners())

### fireVetoableChange

protected void **fireVetoableChange**([String](http://docs.google.com/java/lang/String.html) propertyName,  
 [Object](http://docs.google.com/java/lang/Object.html) oldValue,  
 [Object](http://docs.google.com/java/lang/Object.html) newValue)  
 throws [PropertyVetoException](http://docs.google.com/java/beans/PropertyVetoException.html)

Fires a PropertyChangeEvent in response to a change in a vetoable property. The event will be delivered to all registered VetoableChangeListeners. If a VetoableChangeListener throws a PropertyVetoException, a new event is fired reverting all VetoableChangeListeners to the old value and the exception is then rethrown. No event will be delivered if oldValue and newValue are the same.

**Parameters:**propertyName - the name of the property that has changedoldValue - the property's previous valuenewValue - the property's new value **Throws:** [PropertyVetoException](http://docs.google.com/java/beans/PropertyVetoException.html) - if a VetoableChangeListener threw PropertyVetoException

### addKeyEventDispatcher

public void **addKeyEventDispatcher**([KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) dispatcher)

Adds a KeyEventDispatcher to this KeyboardFocusManager's dispatcher chain. This KeyboardFocusManager will request that each KeyEventDispatcher dispatch KeyEvents generated by the user before finally dispatching the KeyEvent itself. KeyEventDispatchers will be notified in the order in which they were added. Notifications will halt as soon as one KeyEventDispatcher returns true from its dispatchKeyEvent method. There is no limit to the total number of KeyEventDispatchers which can be added, nor to the number of times which a particular KeyEventDispatcher instance can be added.

If a null dispatcher is specified, no action is taken and no exception is thrown.

In a multithreaded application, [KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) behaves the same as other AWT listeners. See [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for more details.

**Parameters:**dispatcher - the KeyEventDispatcher to add to the dispatcher chain**See Also:**[removeKeyEventDispatcher(java.awt.KeyEventDispatcher)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeKeyEventDispatcher(java.awt.KeyEventDispatcher))

### removeKeyEventDispatcher

public void **removeKeyEventDispatcher**([KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) dispatcher)

Removes a KeyEventDispatcher which was previously added to this KeyboardFocusManager's dispatcher chain. This KeyboardFocusManager cannot itself be removed, unless it was explicitly re-registered via a call to addKeyEventDispatcher.

If a null dispatcher is specified, if the specified dispatcher is not in the dispatcher chain, or if this KeyboardFocusManager is specified without having been explicitly re-registered, no action is taken and no exception is thrown.

In a multithreaded application, [KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) behaves the same as other AWT listeners. See [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for more details.

**Parameters:**dispatcher - the KeyEventDispatcher to remove from the dispatcher chain**See Also:**[addKeyEventDispatcher(java.awt.KeyEventDispatcher)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addKeyEventDispatcher(java.awt.KeyEventDispatcher))

### getKeyEventDispatchers

protected [List](http://docs.google.com/java/util/List.html)<[KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html)> **getKeyEventDispatchers**()

Returns this KeyboardFocusManager's KeyEventDispatcher chain as a List. The List will not include this KeyboardFocusManager unless it was explicitly re-registered via a call to addKeyEventDispatcher. If no other KeyEventDispatchers are registered, implementations are free to return null or a List of length 0. Client code should not assume one behavior over another, nor should it assume that the behavior, once established, will not change.

**Returns:**a possibly null or empty List of KeyEventDispatchers**See Also:**[addKeyEventDispatcher(java.awt.KeyEventDispatcher)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addKeyEventDispatcher(java.awt.KeyEventDispatcher)), [removeKeyEventDispatcher(java.awt.KeyEventDispatcher)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeKeyEventDispatcher(java.awt.KeyEventDispatcher))

### addKeyEventPostProcessor

public void **addKeyEventPostProcessor**([KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) processor)

Adds a KeyEventPostProcessor to this KeyboardFocusManager's post- processor chain. After a KeyEvent has been dispatched to and handled by its target, KeyboardFocusManager will request that each KeyEventPostProcessor perform any necessary post-processing as part of the KeyEvent's final resolution. KeyEventPostProcessors will be notified in the order in which they were added; the current KeyboardFocusManager will be notified last. Notifications will halt as soon as one KeyEventPostProcessor returns true from its postProcessKeyEvent method. There is no limit to the the total number of KeyEventPostProcessors that can be added, nor to the number of times that a particular KeyEventPostProcessor instance can be added.

If a null post-processor is specified, no action is taken and no exception is thrown.

In a multithreaded application, [KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) behaves the same as other AWT listeners. See [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for more details.

**Parameters:**processor - the KeyEventPostProcessor to add to the post-processor chain**See Also:**[removeKeyEventPostProcessor(java.awt.KeyEventPostProcessor)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeKeyEventPostProcessor(java.awt.KeyEventPostProcessor))

### removeKeyEventPostProcessor

public void **removeKeyEventPostProcessor**([KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) processor)

Removes a previously added KeyEventPostProcessor from this KeyboardFocusManager's post-processor chain. This KeyboardFocusManager cannot itself be entirely removed from the chain. Only additional references added via addKeyEventPostProcessor can be removed.

If a null post-processor is specified, if the specified post-processor is not in the post-processor chain, or if this KeyboardFocusManager is specified without having been explicitly added, no action is taken and no exception is thrown.

In a multithreaded application, [KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) behaves the same as other AWT listeners. See [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for more details.

**Parameters:**processor - the KeyEventPostProcessor to remove from the post- processor chain**See Also:**[addKeyEventPostProcessor(java.awt.KeyEventPostProcessor)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addKeyEventPostProcessor(java.awt.KeyEventPostProcessor))

### getKeyEventPostProcessors

protected [List](http://docs.google.com/java/util/List.html)<[KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html)> **getKeyEventPostProcessors**()

Returns this KeyboardFocusManager's KeyEventPostProcessor chain as a List. The List will not include this KeyboardFocusManager unless it was explicitly added via a call to addKeyEventPostProcessor. If no KeyEventPostProcessors are registered, implementations are free to return null or a List of length 0. Client code should not assume one behavior over another, nor should it assume that the behavior, once established, will not change.

**Returns:**a possibly null or empty List of KeyEventPostProcessors**See Also:**[addKeyEventPostProcessor(java.awt.KeyEventPostProcessor)](http://docs.google.com/java/awt/KeyboardFocusManager.html#addKeyEventPostProcessor(java.awt.KeyEventPostProcessor)), [removeKeyEventPostProcessor(java.awt.KeyEventPostProcessor)](http://docs.google.com/java/awt/KeyboardFocusManager.html#removeKeyEventPostProcessor(java.awt.KeyEventPostProcessor))

### dispatchEvent

public abstract boolean **dispatchEvent**([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) e)

This method is called by the AWT event dispatcher requesting that the current KeyboardFocusManager dispatch the specified event on its behalf. It is expected that all KeyboardFocusManagers will dispatch all FocusEvents, all WindowEvents related to focus, and all KeyEvents. These events should be dispatched based on the KeyboardFocusManager's notion of the focus owner and the focused and active Windows, sometimes overriding the source of the specified AWTEvent. Dispatching must be done using redispatchEvent to prevent the AWT event dispatcher from recursively requesting that the KeyboardFocusManager dispatch the event again. If this method returns false, then the AWT event dispatcher will attempt to dispatch the event itself.

**Parameters:**e - the AWTEvent to be dispatched **Returns:**true if this method dispatched the event; false otherwise**See Also:**[redispatchEvent(java.awt.Component, java.awt.AWTEvent)](http://docs.google.com/java/awt/KeyboardFocusManager.html#redispatchEvent(java.awt.Component,%20java.awt.AWTEvent)), [dispatchKeyEvent(java.awt.event.KeyEvent)](http://docs.google.com/java/awt/KeyboardFocusManager.html#dispatchKeyEvent(java.awt.event.KeyEvent))

### redispatchEvent

public final void **redispatchEvent**([Component](http://docs.google.com/java/awt/Component.html) target,  
 [AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) e)

Redispatches an AWTEvent in such a way that the AWT event dispatcher will not recursively request that the KeyboardFocusManager, or any installed KeyEventDispatchers, dispatch the event again. Client implementations of dispatchEvent and client-defined KeyEventDispatchers must call redispatchEvent(target, e) instead of target.dispatchEvent(e) to dispatch an event.

This method is intended to be used only by KeyboardFocusManagers and KeyEventDispatchers. It is not for general client use.

**Parameters:**target - the Component to which the event should be dispatchede - the event to dispatch**See Also:**[dispatchEvent(java.awt.AWTEvent)](http://docs.google.com/java/awt/KeyboardFocusManager.html#dispatchEvent(java.awt.AWTEvent)), [KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html)

### dispatchKeyEvent

public abstract boolean **dispatchKeyEvent**([KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)

Typically this method will be called by dispatchEvent if no other KeyEventDispatcher in the dispatcher chain dispatched the KeyEvent, or if no other KeyEventDispatchers are registered. If an implementation of this method returns false, dispatchEvent may try to dispatch the KeyEvent itself, or may simply return false. If true is returned, dispatchEvent should return true as well.

**Specified by:**[dispatchKeyEvent](http://docs.google.com/java/awt/KeyEventDispatcher.html#dispatchKeyEvent(java.awt.event.KeyEvent)) in interface [KeyEventDispatcher](http://docs.google.com/java/awt/KeyEventDispatcher.html) **Parameters:**e - the KeyEvent which the current KeyboardFocusManager has requested that this KeyEventDispatcher dispatch **Returns:**true if the KeyEvent was dispatched; false otherwise**See Also:**[dispatchEvent(java.awt.AWTEvent)](http://docs.google.com/java/awt/KeyboardFocusManager.html#dispatchEvent(java.awt.AWTEvent))

### postProcessKeyEvent

public abstract boolean **postProcessKeyEvent**([KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html) e)

This method will be called by dispatchKeyEvent. By default, this method will handle any unconsumed KeyEvents that map to an AWT MenuShortcut by consuming the event and activating the shortcut.

**Specified by:**[postProcessKeyEvent](http://docs.google.com/java/awt/KeyEventPostProcessor.html#postProcessKeyEvent(java.awt.event.KeyEvent)) in interface [KeyEventPostProcessor](http://docs.google.com/java/awt/KeyEventPostProcessor.html) **Parameters:**e - the KeyEvent to post-process **Returns:**true to indicate that no other KeyEventPostProcessor will be notified of the KeyEvent.**See Also:**[dispatchKeyEvent(java.awt.event.KeyEvent)](http://docs.google.com/java/awt/KeyboardFocusManager.html#dispatchKeyEvent(java.awt.event.KeyEvent)), [MenuShortcut](http://docs.google.com/java/awt/MenuShortcut.html)

### processKeyEvent

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

This method initiates a focus traversal operation if and only if the KeyEvent represents a focus traversal key for the specified focusedComponent. It is expected that focusedComponent is the current focus owner, although this need not be the case. If it is not, focus traversal will nevertheless proceed as if focusedComponent were the current focus owner.

**Parameters:**focusedComponent - the Component that will be the basis for a focus traversal operation if the specified event represents a focus traversal key for the Componente - the event that may represent a focus traversal key

### enqueueKeyEvents

protected abstract void **enqueueKeyEvents**(long after,  
 [Component](http://docs.google.com/java/awt/Component.html) untilFocused)

Called by the AWT to notify the KeyboardFocusManager that it should delay dispatching of KeyEvents until the specified Component becomes the focus owner. If client code requests a focus change, and the AWT determines that this request might be granted by the native windowing system, then the AWT will call this method. It is the responsibility of the KeyboardFocusManager to delay dispatching of KeyEvents with timestamps later than the specified time stamp until the specified Component receives a FOCUS\_GAINED event, or the AWT cancels the delay request by invoking dequeueKeyEvents or discardKeyEvents.

**Parameters:**after - timestamp of current event, or the current, system time if the current event has no timestamp, or the AWT cannot determine which event is currently being handleduntilFocused - Component which should receive a FOCUS\_GAINED event before any pending KeyEvents**See Also:**[dequeueKeyEvents(long, java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#dequeueKeyEvents(long,%20java.awt.Component)), [discardKeyEvents(java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#discardKeyEvents(java.awt.Component))

### dequeueKeyEvents

protected abstract void **dequeueKeyEvents**(long after,  
 [Component](http://docs.google.com/java/awt/Component.html) untilFocused)

Called by the AWT to notify the KeyboardFocusManager that it should cancel delayed dispatching of KeyEvents. All KeyEvents which were enqueued because of a call to enqueueKeyEvents with the same timestamp and Component should be released for normal dispatching to the current focus owner. If the given timestamp is less than zero, the outstanding enqueue request for the given Component with the  **oldest** timestamp (if any) should be cancelled.

**Parameters:**after - the timestamp specified in the call to enqueueKeyEvents, or any value < 0untilFocused - the Component specified in the call to enqueueKeyEvents**See Also:**[enqueueKeyEvents(long, java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#enqueueKeyEvents(long,%20java.awt.Component)), [discardKeyEvents(java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#discardKeyEvents(java.awt.Component))

### discardKeyEvents

protected abstract void **discardKeyEvents**([Component](http://docs.google.com/java/awt/Component.html) comp)

Called by the AWT to notify the KeyboardFocusManager that it should cancel delayed dispatching of KeyEvents. All KeyEvents which were enqueued because of one or more calls to enqueueKeyEvents with the same Component should be discarded.

**Parameters:**comp - the Component specified in one or more calls to enqueueKeyEvents**See Also:**[enqueueKeyEvents(long, java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#enqueueKeyEvents(long,%20java.awt.Component)), [dequeueKeyEvents(long, java.awt.Component)](http://docs.google.com/java/awt/KeyboardFocusManager.html#dequeueKeyEvents(long,%20java.awt.Component))

### focusNextComponent

public abstract void **focusNextComponent**([Component](http://docs.google.com/java/awt/Component.html) aComponent)

Focuses the Component after aComponent, typically based on a FocusTraversalPolicy.

**Parameters:**aComponent - the Component that is the basis for the focus traversal operation**See Also:**[FocusTraversalPolicy](http://docs.google.com/java/awt/FocusTraversalPolicy.html)

### focusPreviousComponent

public abstract void **focusPreviousComponent**([Component](http://docs.google.com/java/awt/Component.html) aComponent)

Focuses the Component before aComponent, typically based on a FocusTraversalPolicy.

**Parameters:**aComponent - the Component that is the basis for the focus traversal operation**See Also:**[FocusTraversalPolicy](http://docs.google.com/java/awt/FocusTraversalPolicy.html)

### upFocusCycle

public abstract void **upFocusCycle**([Component](http://docs.google.com/java/awt/Component.html) aComponent)

Moves the focus up one focus traversal cycle. Typically, the focus owner is set to aComponent's focus cycle root, and the current focus cycle root is set to the new focus owner's focus cycle root. If, however, aComponent's focus cycle root is a Window, then typically the focus owner is set to the Window's default Component to focus, and the current focus cycle root is unchanged.

**Parameters:**aComponent - the Component that is the basis for the focus traversal operation

### downFocusCycle

public abstract void **downFocusCycle**([Container](http://docs.google.com/java/awt/Container.html) aContainer)

Moves the focus down one focus traversal cycle. Typically, if aContainer is a focus cycle root, then the focus owner is set to aContainer's default Component to focus, and the current focus cycle root is set to aContainer. If aContainer is not a focus cycle root, then no focus traversal operation occurs.

**Parameters:**aContainer - the Container that is the basis for the focus traversal operation

### focusNextComponent

public final void **focusNextComponent**()

Focuses the Component after the current focus owner.

### focusPreviousComponent

public final void **focusPreviousComponent**()

Focuses the Component before the current focus owner.

### upFocusCycle

public final void **upFocusCycle**()

Moves the focus up one focus traversal cycle from the current focus owner. Typically, the new focus owner is set to the current focus owner's focus cycle root, and the current focus cycle root is set to the new focus owner's focus cycle root. If, however, the current focus owner's focus cycle root is a Window, then typically the focus owner is set to the focus cycle root's default Component to focus, and the current focus cycle root is unchanged.

### downFocusCycle

public final void **downFocusCycle**()

Moves the focus down one focus traversal cycle from the current focus owner, if and only if the current focus owner is a Container that is a focus cycle root. Typically, the focus owner is set to the current focus owner's default Component to focus, and the current focus cycle root is set to the current focus owner. If the current focus owner is not a Container that is a focus cycle root, then no focus traversal operation occurs.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyboardFocusManager.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/java/awt/JobAttributes.SidesType.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/KeyEventDispatcher.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/KeyboardFocusManager.html)    [**NO FRAMES**](http://docs.google.com/KeyboardFocusManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#35nkun2) |

[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).