| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/List.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/LinearGradientPaint.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/List.AccessibleAWTList.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/List.html)    [**NO FRAMES**](http://docs.google.com/List.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#3dy6vkm) | [CONSTR](#4d34og8) | [METHOD](#2s8eyo1) | DETAIL: FIELD | [CONSTR](#26in1rg) | [METHOD](#44sinio) |

## **java.awt**

Class List

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

**All Implemented Interfaces:** [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html), [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html), [MenuContainer](http://docs.google.com/java/awt/MenuContainer.html), [Serializable](http://docs.google.com/java/io/Serializable.html), [Accessible](http://docs.google.com/javax/accessibility/Accessible.html)

public class **List**extends [Component](http://docs.google.com/java/awt/Component.html)implements [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html), [Accessible](http://docs.google.com/javax/accessibility/Accessible.html)

The List component presents the user with a scrolling list of text items. The list can be set up so that the user can choose either one item or multiple items.

For example, the code . . .

List lst = new List(4, false);  
 lst.add("Mercury");  
 lst.add("Venus");  
 lst.add("Earth");  
 lst.add("JavaSoft");  
 lst.add("Mars");  
 lst.add("Jupiter");  
 lst.add("Saturn");  
 lst.add("Uranus");  
 lst.add("Neptune");  
 lst.add("Pluto");  
 cnt.add(lst);

where cnt is a container, produces the following scrolling list:



If the List allows multiple selections, then clicking on an item that is already selected deselects it. In the preceding example, only one item from the scrolling list can be selected at a time, since the second argument when creating the new scrolling list is false. If the List does not allow multiple selections, selecting an item causes any other selected item to be deselected.

Note that the list in the example shown was created with four visible rows. Once the list has been created, the number of visible rows cannot be changed. A default List is created with four rows, so that lst = new List() is equivalent to list = new List(4, false).

Beginning with Java 1.1, the Abstract Window Toolkit sends the List object all mouse, keyboard, and focus events that occur over it. (The old AWT event model is being maintained only for backwards compatibility, and its use is discouraged.)

When an item is selected or deselected by the user, AWT sends an instance of ItemEvent to the list. When the user double-clicks on an item in a scrolling list, AWT sends an instance of ActionEvent to the list following the item event. AWT also generates an action event when the user presses the return key while an item in the list is selected.

If an application wants to perform some action based on an item in this list being selected or activated by the user, it should implement ItemListener or ActionListener as appropriate and register the new listener to receive events from this list.

For multiple-selection scrolling lists, it is considered a better user interface to use an external gesture (such as clicking on a button) to trigger the action.

**Since:** JDK1.0 **See Also:**[ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html), [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html), [ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html), [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.awt.List)

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**List.AccessibleAWTList**](http://docs.google.com/java/awt/List.AccessibleAWTList.html)            This class implements accessibility support for the List class. |

| **Nested classes/interfaces inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [Component.AccessibleAWTComponent](http://docs.google.com/java/awt/Component.AccessibleAWTComponent.html), [Component.BaselineResizeBehavior](http://docs.google.com/java/awt/Component.BaselineResizeBehavior.html), [Component.BltBufferStrategy](http://docs.google.com/java/awt/Component.BltBufferStrategy.html), [Component.FlipBufferStrategy](http://docs.google.com/java/awt/Component.FlipBufferStrategy.html) |

| **Field Summary** | |
| --- | --- |

| **Fields inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [BOTTOM\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#BOTTOM_ALIGNMENT), [CENTER\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#CENTER_ALIGNMENT), [LEFT\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#LEFT_ALIGNMENT), [RIGHT\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#RIGHT_ALIGNMENT), [TOP\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#TOP_ALIGNMENT) |

| **Fields inherited from interface java.awt.image.**[**ImageObserver**](http://docs.google.com/java/awt/image/ImageObserver.html) |
| --- |
| [ABORT](http://docs.google.com/java/awt/image/ImageObserver.html#ABORT), [ALLBITS](http://docs.google.com/java/awt/image/ImageObserver.html#ALLBITS), [ERROR](http://docs.google.com/java/awt/image/ImageObserver.html#ERROR), [FRAMEBITS](http://docs.google.com/java/awt/image/ImageObserver.html#FRAMEBITS), [HEIGHT](http://docs.google.com/java/awt/image/ImageObserver.html#HEIGHT), [PROPERTIES](http://docs.google.com/java/awt/image/ImageObserver.html#PROPERTIES), [SOMEBITS](http://docs.google.com/java/awt/image/ImageObserver.html#SOMEBITS), [WIDTH](http://docs.google.com/java/awt/image/ImageObserver.html#WIDTH) |

| **Constructor Summary** | |
| --- | --- |
| [**List**](http://docs.google.com/java/awt/List.html#List())()            Creates a new scrolling list. |
| [**List**](http://docs.google.com/java/awt/List.html#List(int))(int rows)            Creates a new scrolling list initialized with the specified number of visible lines. |
| [**List**](http://docs.google.com/java/awt/List.html#List(int,%20boolean))(int rows, boolean multipleMode)            Creates a new scrolling list initialized to display the specified number of rows. |

| **Method Summary** | |
| --- | --- |
| void | [**add**](http://docs.google.com/java/awt/List.html#add(java.lang.String))([String](http://docs.google.com/java/lang/String.html) item)            Adds the specified item to the end of scrolling list. |
| void | [**add**](http://docs.google.com/java/awt/List.html#add(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) item, int index)            Adds the specified item to the the scrolling list at the position indicated by the index. |
| void | [**addActionListener**](http://docs.google.com/java/awt/List.html#addActionListener(java.awt.event.ActionListener))([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)            Adds the specified action listener to receive action events from this list. |
| void | [**addItem**](http://docs.google.com/java/awt/List.html#addItem(java.lang.String))([String](http://docs.google.com/java/lang/String.html) item)  **Deprecated.** *replaced by add(String).* |
| void | [**addItem**](http://docs.google.com/java/awt/List.html#addItem(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) item, int index)  **Deprecated.** *replaced by add(String, int).* |
| void | [**addItemListener**](http://docs.google.com/java/awt/List.html#addItemListener(java.awt.event.ItemListener))([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)            Adds the specified item listener to receive item events from this list. |
| void | [**addNotify**](http://docs.google.com/java/awt/List.html#addNotify())()            Creates the peer for the list. |
| boolean | [**allowsMultipleSelections**](http://docs.google.com/java/awt/List.html#allowsMultipleSelections())()  **Deprecated.** *As of JDK version 1.1, replaced by isMultipleMode().* |
| void | [**clear**](http://docs.google.com/java/awt/List.html#clear())()  **Deprecated.** *As of JDK version 1.1, replaced by removeAll().* |
| int | [**countItems**](http://docs.google.com/java/awt/List.html#countItems())()  **Deprecated.** *As of JDK version 1.1, replaced by getItemCount().* |
| void | [**delItem**](http://docs.google.com/java/awt/List.html#delItem(int))(int position)  **Deprecated.** *replaced by remove(String) and remove(int).* |
| void | [**delItems**](http://docs.google.com/java/awt/List.html#delItems(int,%20int))(int start, int end)  **Deprecated.** *As of JDK version 1.1, Not for public use in the future. This method is expected to be retained only as a package private method.* |
| void | [**deselect**](http://docs.google.com/java/awt/List.html#deselect(int))(int index)            Deselects the item at the specified index. |
| [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) | [**getAccessibleContext**](http://docs.google.com/java/awt/List.html#getAccessibleContext())()            Gets the AccessibleContext associated with this List. |
| [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)[] | [**getActionListeners**](http://docs.google.com/java/awt/List.html#getActionListeners())()            Returns an array of all the action listeners registered on this list. |
| [String](http://docs.google.com/java/lang/String.html) | [**getItem**](http://docs.google.com/java/awt/List.html#getItem(int))(int index)            Gets the item associated with the specified index. |
| int | [**getItemCount**](http://docs.google.com/java/awt/List.html#getItemCount())()            Gets the number of items in the list. |
| [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)[] | [**getItemListeners**](http://docs.google.com/java/awt/List.html#getItemListeners())()            Returns an array of all the item listeners registered on this list. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getItems**](http://docs.google.com/java/awt/List.html#getItems())()            Gets the items in the list. |
| | <T extends [EventListener](http://docs.google.com/java/util/EventListener.html)>  T[] | | --- | | [**getListeners**](http://docs.google.com/java/awt/List.html#getListeners(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<T> listenerType)            Returns an array of all the objects currently registered as *Foo*Listeners upon this List. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getMinimumSize**](http://docs.google.com/java/awt/List.html#getMinimumSize())()            Determines the minimum size of this scrolling list. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getMinimumSize**](http://docs.google.com/java/awt/List.html#getMinimumSize(int))(int rows)            Gets the minumum dimensions for a list with the specified number of rows. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredSize**](http://docs.google.com/java/awt/List.html#getPreferredSize())()            Gets the preferred size of this scrolling list. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredSize**](http://docs.google.com/java/awt/List.html#getPreferredSize(int))(int rows)            Gets the preferred dimensions for a list with the specified number of rows. |
| int | [**getRows**](http://docs.google.com/java/awt/List.html#getRows())()            Gets the number of visible lines in this list. |
| int | [**getSelectedIndex**](http://docs.google.com/java/awt/List.html#getSelectedIndex())()            Gets the index of the selected item on the list, |
| int[] | [**getSelectedIndexes**](http://docs.google.com/java/awt/List.html#getSelectedIndexes())()            Gets the selected indexes on the list. |
| [String](http://docs.google.com/java/lang/String.html) | [**getSelectedItem**](http://docs.google.com/java/awt/List.html#getSelectedItem())()            Gets the selected item on this scrolling list. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getSelectedItems**](http://docs.google.com/java/awt/List.html#getSelectedItems())()            Gets the selected items on this scrolling list. |
| [Object](http://docs.google.com/java/lang/Object.html)[] | [**getSelectedObjects**](http://docs.google.com/java/awt/List.html#getSelectedObjects())()            Gets the selected items on this scrolling list in an array of Objects. |
| int | [**getVisibleIndex**](http://docs.google.com/java/awt/List.html#getVisibleIndex())()            Gets the index of the item that was last made visible by the method makeVisible. |
| boolean | [**isIndexSelected**](http://docs.google.com/java/awt/List.html#isIndexSelected(int))(int index)            Determines if the specified item in this scrolling list is selected. |
| boolean | [**isMultipleMode**](http://docs.google.com/java/awt/List.html#isMultipleMode())()            Determines whether this list allows multiple selections. |
| boolean | [**isSelected**](http://docs.google.com/java/awt/List.html#isSelected(int))(int index)  **Deprecated.** *As of JDK version 1.1, replaced by isIndexSelected(int).* |
| void | [**makeVisible**](http://docs.google.com/java/awt/List.html#makeVisible(int))(int index)            Makes the item at the specified index visible. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**minimumSize**](http://docs.google.com/java/awt/List.html#minimumSize())()  **Deprecated.** *As of JDK version 1.1, replaced by getMinimumSize().* |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**minimumSize**](http://docs.google.com/java/awt/List.html#minimumSize(int))(int rows)  **Deprecated.** *As of JDK version 1.1, replaced by getMinimumSize(int).* |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**paramString**](http://docs.google.com/java/awt/List.html#paramString())()            Returns the parameter string representing the state of this scrolling list. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**preferredSize**](http://docs.google.com/java/awt/List.html#preferredSize())()  **Deprecated.** *As of JDK version 1.1, replaced by getPreferredSize().* |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**preferredSize**](http://docs.google.com/java/awt/List.html#preferredSize(int))(int rows)  **Deprecated.** *As of JDK version 1.1, replaced by getPreferredSize(int).* |
| protected  void | [**processActionEvent**](http://docs.google.com/java/awt/List.html#processActionEvent(java.awt.event.ActionEvent))([ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html) e)            Processes action events occurring on this component by dispatching them to any registered ActionListener objects. |
| protected  void | [**processEvent**](http://docs.google.com/java/awt/List.html#processEvent(java.awt.AWTEvent))([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) e)            Processes events on this scrolling list. |
| protected  void | [**processItemEvent**](http://docs.google.com/java/awt/List.html#processItemEvent(java.awt.event.ItemEvent))([ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html) e)            Processes item events occurring on this list by dispatching them to any registered ItemListener objects. |
| void | [**remove**](http://docs.google.com/java/awt/List.html#remove(int))(int position)            Removes the item at the specified position from this scrolling list. |
| void | [**remove**](http://docs.google.com/java/awt/List.html#remove(java.lang.String))([String](http://docs.google.com/java/lang/String.html) item)            Removes the first occurrence of an item from the list. |
| void | [**removeActionListener**](http://docs.google.com/java/awt/List.html#removeActionListener(java.awt.event.ActionListener))([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)            Removes the specified action listener so that it no longer receives action events from this list. |
| void | [**removeAll**](http://docs.google.com/java/awt/List.html#removeAll())()            Removes all items from this list. |
| void | [**removeItemListener**](http://docs.google.com/java/awt/List.html#removeItemListener(java.awt.event.ItemListener))([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)            Removes the specified item listener so that it no longer receives item events from this list. |
| void | [**removeNotify**](http://docs.google.com/java/awt/List.html#removeNotify())()            Removes the peer for this list. |
| void | [**replaceItem**](http://docs.google.com/java/awt/List.html#replaceItem(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) newValue, int index)            Replaces the item at the specified index in the scrolling list with the new string. |
| void | [**select**](http://docs.google.com/java/awt/List.html#select(int))(int index)            Selects the item at the specified index in the scrolling list. |
| void | [**setMultipleMode**](http://docs.google.com/java/awt/List.html#setMultipleMode(boolean))(boolean b)            Sets the flag that determines whether this list allows multiple selections. |
| void | [**setMultipleSelections**](http://docs.google.com/java/awt/List.html#setMultipleSelections(boolean))(boolean b)  **Deprecated.** *As of JDK version 1.1, replaced by setMultipleMode(boolean).* |

| **Methods inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [action](http://docs.google.com/java/awt/Component.html#action(java.awt.Event,%20java.lang.Object)), [add](http://docs.google.com/java/awt/Component.html#add(java.awt.PopupMenu)), [addComponentListener](http://docs.google.com/java/awt/Component.html#addComponentListener(java.awt.event.ComponentListener)), [addFocusListener](http://docs.google.com/java/awt/Component.html#addFocusListener(java.awt.event.FocusListener)), [addHierarchyBoundsListener](http://docs.google.com/java/awt/Component.html#addHierarchyBoundsListener(java.awt.event.HierarchyBoundsListener)), [addHierarchyListener](http://docs.google.com/java/awt/Component.html#addHierarchyListener(java.awt.event.HierarchyListener)), [addInputMethodListener](http://docs.google.com/java/awt/Component.html#addInputMethodListener(java.awt.event.InputMethodListener)), [addKeyListener](http://docs.google.com/java/awt/Component.html#addKeyListener(java.awt.event.KeyListener)), [addMouseListener](http://docs.google.com/java/awt/Component.html#addMouseListener(java.awt.event.MouseListener)), [addMouseMotionListener](http://docs.google.com/java/awt/Component.html#addMouseMotionListener(java.awt.event.MouseMotionListener)), [addMouseWheelListener](http://docs.google.com/java/awt/Component.html#addMouseWheelListener(java.awt.event.MouseWheelListener)), [addPropertyChangeListener](http://docs.google.com/java/awt/Component.html#addPropertyChangeListener(java.beans.PropertyChangeListener)), [addPropertyChangeListener](http://docs.google.com/java/awt/Component.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [applyComponentOrientation](http://docs.google.com/java/awt/Component.html#applyComponentOrientation(java.awt.ComponentOrientation)), [areFocusTraversalKeysSet](http://docs.google.com/java/awt/Component.html#areFocusTraversalKeysSet(int)), [bounds](http://docs.google.com/java/awt/Component.html#bounds()), [checkImage](http://docs.google.com/java/awt/Component.html#checkImage(java.awt.Image,%20java.awt.image.ImageObserver)), [checkImage](http://docs.google.com/java/awt/Component.html#checkImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [coalesceEvents](http://docs.google.com/java/awt/Component.html#coalesceEvents(java.awt.AWTEvent,%20java.awt.AWTEvent)), [contains](http://docs.google.com/java/awt/Component.html#contains(int,%20int)), [contains](http://docs.google.com/java/awt/Component.html#contains(java.awt.Point)), [createImage](http://docs.google.com/java/awt/Component.html#createImage(java.awt.image.ImageProducer)), [createImage](http://docs.google.com/java/awt/Component.html#createImage(int,%20int)), [createVolatileImage](http://docs.google.com/java/awt/Component.html#createVolatileImage(int,%20int)), [createVolatileImage](http://docs.google.com/java/awt/Component.html#createVolatileImage(int,%20int,%20java.awt.ImageCapabilities)), [deliverEvent](http://docs.google.com/java/awt/Component.html#deliverEvent(java.awt.Event)), [disable](http://docs.google.com/java/awt/Component.html#disable()), [disableEvents](http://docs.google.com/java/awt/Component.html#disableEvents(long)), [dispatchEvent](http://docs.google.com/java/awt/Component.html#dispatchEvent(java.awt.AWTEvent)), [doLayout](http://docs.google.com/java/awt/Component.html#doLayout()), [enable](http://docs.google.com/java/awt/Component.html#enable()), [enable](http://docs.google.com/java/awt/Component.html#enable(boolean)), [enableEvents](http://docs.google.com/java/awt/Component.html#enableEvents(long)), [enableInputMethods](http://docs.google.com/java/awt/Component.html#enableInputMethods(boolean)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20boolean,%20boolean)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20byte,%20byte)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20char,%20char)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20double,%20double)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20float,%20float)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20int,%20int)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20long,%20long)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20java.lang.Object,%20java.lang.Object)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20short,%20short)), [getAlignmentX](http://docs.google.com/java/awt/Component.html#getAlignmentX()), [getAlignmentY](http://docs.google.com/java/awt/Component.html#getAlignmentY()), [getBackground](http://docs.google.com/java/awt/Component.html#getBackground()), [getBaseline](http://docs.google.com/java/awt/Component.html#getBaseline(int,%20int)), [getBaselineResizeBehavior](http://docs.google.com/java/awt/Component.html#getBaselineResizeBehavior()), [getBounds](http://docs.google.com/java/awt/Component.html#getBounds()), [getBounds](http://docs.google.com/java/awt/Component.html#getBounds(java.awt.Rectangle)), [getColorModel](http://docs.google.com/java/awt/Component.html#getColorModel()), [getComponentAt](http://docs.google.com/java/awt/Component.html#getComponentAt(int,%20int)), [getComponentAt](http://docs.google.com/java/awt/Component.html#getComponentAt(java.awt.Point)), [getComponentListeners](http://docs.google.com/java/awt/Component.html#getComponentListeners()), [getComponentOrientation](http://docs.google.com/java/awt/Component.html#getComponentOrientation()), [getCursor](http://docs.google.com/java/awt/Component.html#getCursor()), [getDropTarget](http://docs.google.com/java/awt/Component.html#getDropTarget()), [getFocusCycleRootAncestor](http://docs.google.com/java/awt/Component.html#getFocusCycleRootAncestor()), [getFocusListeners](http://docs.google.com/java/awt/Component.html#getFocusListeners()), [getFocusTraversalKeys](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeys(int)), [getFocusTraversalKeysEnabled](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeysEnabled()), [getFont](http://docs.google.com/java/awt/Component.html#getFont()), [getFontMetrics](http://docs.google.com/java/awt/Component.html#getFontMetrics(java.awt.Font)), [getForeground](http://docs.google.com/java/awt/Component.html#getForeground()), [getGraphics](http://docs.google.com/java/awt/Component.html#getGraphics()), [getGraphicsConfiguration](http://docs.google.com/java/awt/Component.html#getGraphicsConfiguration()), [getHeight](http://docs.google.com/java/awt/Component.html#getHeight()), [getHierarchyBoundsListeners](http://docs.google.com/java/awt/Component.html#getHierarchyBoundsListeners()), [getHierarchyListeners](http://docs.google.com/java/awt/Component.html#getHierarchyListeners()), [getIgnoreRepaint](http://docs.google.com/java/awt/Component.html#getIgnoreRepaint()), [getInputContext](http://docs.google.com/java/awt/Component.html#getInputContext()), [getInputMethodListeners](http://docs.google.com/java/awt/Component.html#getInputMethodListeners()), [getInputMethodRequests](http://docs.google.com/java/awt/Component.html#getInputMethodRequests()), [getKeyListeners](http://docs.google.com/java/awt/Component.html#getKeyListeners()), [getLocale](http://docs.google.com/java/awt/Component.html#getLocale()), [getLocation](http://docs.google.com/java/awt/Component.html#getLocation()), [getLocation](http://docs.google.com/java/awt/Component.html#getLocation(java.awt.Point)), [getLocationOnScreen](http://docs.google.com/java/awt/Component.html#getLocationOnScreen()), [getMaximumSize](http://docs.google.com/java/awt/Component.html#getMaximumSize()), [getMouseListeners](http://docs.google.com/java/awt/Component.html#getMouseListeners()), [getMouseMotionListeners](http://docs.google.com/java/awt/Component.html#getMouseMotionListeners()), [getMousePosition](http://docs.google.com/java/awt/Component.html#getMousePosition()), [getMouseWheelListeners](http://docs.google.com/java/awt/Component.html#getMouseWheelListeners()), [getName](http://docs.google.com/java/awt/Component.html#getName()), [getParent](http://docs.google.com/java/awt/Component.html#getParent()), [getPeer](http://docs.google.com/java/awt/Component.html#getPeer()), [getPropertyChangeListeners](http://docs.google.com/java/awt/Component.html#getPropertyChangeListeners()), [getPropertyChangeListeners](http://docs.google.com/java/awt/Component.html#getPropertyChangeListeners(java.lang.String)), [getSize](http://docs.google.com/java/awt/Component.html#getSize()), [getSize](http://docs.google.com/java/awt/Component.html#getSize(java.awt.Dimension)), [getToolkit](http://docs.google.com/java/awt/Component.html#getToolkit()), [getTreeLock](http://docs.google.com/java/awt/Component.html#getTreeLock()), [getWidth](http://docs.google.com/java/awt/Component.html#getWidth()), [getX](http://docs.google.com/java/awt/Component.html#getX()), [getY](http://docs.google.com/java/awt/Component.html#getY()), [gotFocus](http://docs.google.com/java/awt/Component.html#gotFocus(java.awt.Event,%20java.lang.Object)), [handleEvent](http://docs.google.com/java/awt/Component.html#handleEvent(java.awt.Event)), [hasFocus](http://docs.google.com/java/awt/Component.html#hasFocus()), [hide](http://docs.google.com/java/awt/Component.html#hide()), [imageUpdate](http://docs.google.com/java/awt/Component.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int)), [inside](http://docs.google.com/java/awt/Component.html#inside(int,%20int)), [invalidate](http://docs.google.com/java/awt/Component.html#invalidate()), [isBackgroundSet](http://docs.google.com/java/awt/Component.html#isBackgroundSet()), [isCursorSet](http://docs.google.com/java/awt/Component.html#isCursorSet()), [isDisplayable](http://docs.google.com/java/awt/Component.html#isDisplayable()), [isDoubleBuffered](http://docs.google.com/java/awt/Component.html#isDoubleBuffered()), [isEnabled](http://docs.google.com/java/awt/Component.html#isEnabled()), [isFocusable](http://docs.google.com/java/awt/Component.html#isFocusable()), [isFocusCycleRoot](http://docs.google.com/java/awt/Component.html#isFocusCycleRoot(java.awt.Container)), [isFocusOwner](http://docs.google.com/java/awt/Component.html#isFocusOwner()), [isFocusTraversable](http://docs.google.com/java/awt/Component.html#isFocusTraversable()), [isFontSet](http://docs.google.com/java/awt/Component.html#isFontSet()), [isForegroundSet](http://docs.google.com/java/awt/Component.html#isForegroundSet()), [isLightweight](http://docs.google.com/java/awt/Component.html#isLightweight()), [isMaximumSizeSet](http://docs.google.com/java/awt/Component.html#isMaximumSizeSet()), [isMinimumSizeSet](http://docs.google.com/java/awt/Component.html#isMinimumSizeSet()), [isOpaque](http://docs.google.com/java/awt/Component.html#isOpaque()), [isPreferredSizeSet](http://docs.google.com/java/awt/Component.html#isPreferredSizeSet()), [isShowing](http://docs.google.com/java/awt/Component.html#isShowing()), [isValid](http://docs.google.com/java/awt/Component.html#isValid()), [isVisible](http://docs.google.com/java/awt/Component.html#isVisible()), [keyDown](http://docs.google.com/java/awt/Component.html#keyDown(java.awt.Event,%20int)), [keyUp](http://docs.google.com/java/awt/Component.html#keyUp(java.awt.Event,%20int)), [layout](http://docs.google.com/java/awt/Component.html#layout()), [list](http://docs.google.com/java/awt/Component.html#list()), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintStream)), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintStream,%20int)), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintWriter)), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintWriter,%20int)), [locate](http://docs.google.com/java/awt/Component.html#locate(int,%20int)), [location](http://docs.google.com/java/awt/Component.html#location()), [lostFocus](http://docs.google.com/java/awt/Component.html#lostFocus(java.awt.Event,%20java.lang.Object)), [mouseDown](http://docs.google.com/java/awt/Component.html#mouseDown(java.awt.Event,%20int,%20int)), [mouseDrag](http://docs.google.com/java/awt/Component.html#mouseDrag(java.awt.Event,%20int,%20int)), [mouseEnter](http://docs.google.com/java/awt/Component.html#mouseEnter(java.awt.Event,%20int,%20int)), [mouseExit](http://docs.google.com/java/awt/Component.html#mouseExit(java.awt.Event,%20int,%20int)), [mouseMove](http://docs.google.com/java/awt/Component.html#mouseMove(java.awt.Event,%20int,%20int)), [mouseUp](http://docs.google.com/java/awt/Component.html#mouseUp(java.awt.Event,%20int,%20int)), [move](http://docs.google.com/java/awt/Component.html#move(int,%20int)), [nextFocus](http://docs.google.com/java/awt/Component.html#nextFocus()), [paint](http://docs.google.com/java/awt/Component.html#paint(java.awt.Graphics)), [paintAll](http://docs.google.com/java/awt/Component.html#paintAll(java.awt.Graphics)), [postEvent](http://docs.google.com/java/awt/Component.html#postEvent(java.awt.Event)), [prepareImage](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20java.awt.image.ImageObserver)), [prepareImage](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [print](http://docs.google.com/java/awt/Component.html#print(java.awt.Graphics)), [printAll](http://docs.google.com/java/awt/Component.html#printAll(java.awt.Graphics)), [processComponentEvent](http://docs.google.com/java/awt/Component.html#processComponentEvent(java.awt.event.ComponentEvent)), [processFocusEvent](http://docs.google.com/java/awt/Component.html#processFocusEvent(java.awt.event.FocusEvent)), [processHierarchyBoundsEvent](http://docs.google.com/java/awt/Component.html#processHierarchyBoundsEvent(java.awt.event.HierarchyEvent)), [processHierarchyEvent](http://docs.google.com/java/awt/Component.html#processHierarchyEvent(java.awt.event.HierarchyEvent)), [processInputMethodEvent](http://docs.google.com/java/awt/Component.html#processInputMethodEvent(java.awt.event.InputMethodEvent)), [processKeyEvent](http://docs.google.com/java/awt/Component.html#processKeyEvent(java.awt.event.KeyEvent)), [processMouseEvent](http://docs.google.com/java/awt/Component.html#processMouseEvent(java.awt.event.MouseEvent)), [processMouseMotionEvent](http://docs.google.com/java/awt/Component.html#processMouseMotionEvent(java.awt.event.MouseEvent)), [processMouseWheelEvent](http://docs.google.com/java/awt/Component.html#processMouseWheelEvent(java.awt.event.MouseWheelEvent)), [remove](http://docs.google.com/java/awt/Component.html#remove(java.awt.MenuComponent)), [removeComponentListener](http://docs.google.com/java/awt/Component.html#removeComponentListener(java.awt.event.ComponentListener)), [removeFocusListener](http://docs.google.com/java/awt/Component.html#removeFocusListener(java.awt.event.FocusListener)), [removeHierarchyBoundsListener](http://docs.google.com/java/awt/Component.html#removeHierarchyBoundsListener(java.awt.event.HierarchyBoundsListener)), [removeHierarchyListener](http://docs.google.com/java/awt/Component.html#removeHierarchyListener(java.awt.event.HierarchyListener)), [removeInputMethodListener](http://docs.google.com/java/awt/Component.html#removeInputMethodListener(java.awt.event.InputMethodListener)), [removeKeyListener](http://docs.google.com/java/awt/Component.html#removeKeyListener(java.awt.event.KeyListener)), [removeMouseListener](http://docs.google.com/java/awt/Component.html#removeMouseListener(java.awt.event.MouseListener)), [removeMouseMotionListener](http://docs.google.com/java/awt/Component.html#removeMouseMotionListener(java.awt.event.MouseMotionListener)), [removeMouseWheelListener](http://docs.google.com/java/awt/Component.html#removeMouseWheelListener(java.awt.event.MouseWheelListener)), [removePropertyChangeListener](http://docs.google.com/java/awt/Component.html#removePropertyChangeListener(java.beans.PropertyChangeListener)), [removePropertyChangeListener](http://docs.google.com/java/awt/Component.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [repaint](http://docs.google.com/java/awt/Component.html#repaint()), [repaint](http://docs.google.com/java/awt/Component.html#repaint(int,%20int,%20int,%20int)), [repaint](http://docs.google.com/java/awt/Component.html#repaint(long)), [repaint](http://docs.google.com/java/awt/Component.html#repaint(long,%20int,%20int,%20int,%20int)), [requestFocus](http://docs.google.com/java/awt/Component.html#requestFocus()), [requestFocus](http://docs.google.com/java/awt/Component.html#requestFocus(boolean)), [requestFocusInWindow](http://docs.google.com/java/awt/Component.html#requestFocusInWindow()), [requestFocusInWindow](http://docs.google.com/java/awt/Component.html#requestFocusInWindow(boolean)), [reshape](http://docs.google.com/java/awt/Component.html#reshape(int,%20int,%20int,%20int)), [resize](http://docs.google.com/java/awt/Component.html#resize(java.awt.Dimension)), [resize](http://docs.google.com/java/awt/Component.html#resize(int,%20int)), [setBackground](http://docs.google.com/java/awt/Component.html#setBackground(java.awt.Color)), [setBounds](http://docs.google.com/java/awt/Component.html#setBounds(int,%20int,%20int,%20int)), [setBounds](http://docs.google.com/java/awt/Component.html#setBounds(java.awt.Rectangle)), [setComponentOrientation](http://docs.google.com/java/awt/Component.html#setComponentOrientation(java.awt.ComponentOrientation)), [setCursor](http://docs.google.com/java/awt/Component.html#setCursor(java.awt.Cursor)), [setDropTarget](http://docs.google.com/java/awt/Component.html#setDropTarget(java.awt.dnd.DropTarget)), [setEnabled](http://docs.google.com/java/awt/Component.html#setEnabled(boolean)), [setFocusable](http://docs.google.com/java/awt/Component.html#setFocusable(boolean)), [setFocusTraversalKeys](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeys(int,%20java.util.Set)), [setFocusTraversalKeysEnabled](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeysEnabled(boolean)), [setFont](http://docs.google.com/java/awt/Component.html#setFont(java.awt.Font)), [setForeground](http://docs.google.com/java/awt/Component.html#setForeground(java.awt.Color)), [setIgnoreRepaint](http://docs.google.com/java/awt/Component.html#setIgnoreRepaint(boolean)), [setLocale](http://docs.google.com/java/awt/Component.html#setLocale(java.util.Locale)), [setLocation](http://docs.google.com/java/awt/Component.html#setLocation(int,%20int)), [setLocation](http://docs.google.com/java/awt/Component.html#setLocation(java.awt.Point)), [setMaximumSize](http://docs.google.com/java/awt/Component.html#setMaximumSize(java.awt.Dimension)), [setMinimumSize](http://docs.google.com/java/awt/Component.html#setMinimumSize(java.awt.Dimension)), [setName](http://docs.google.com/java/awt/Component.html#setName(java.lang.String)), [setPreferredSize](http://docs.google.com/java/awt/Component.html#setPreferredSize(java.awt.Dimension)), [setSize](http://docs.google.com/java/awt/Component.html#setSize(java.awt.Dimension)), [setSize](http://docs.google.com/java/awt/Component.html#setSize(int,%20int)), [setVisible](http://docs.google.com/java/awt/Component.html#setVisible(boolean)), [show](http://docs.google.com/java/awt/Component.html#show()), [show](http://docs.google.com/java/awt/Component.html#show(boolean)), [size](http://docs.google.com/java/awt/Component.html#size()), [toString](http://docs.google.com/java/awt/Component.html#toString()), [transferFocus](http://docs.google.com/java/awt/Component.html#transferFocus()), [transferFocusBackward](http://docs.google.com/java/awt/Component.html#transferFocusBackward()), [transferFocusUpCycle](http://docs.google.com/java/awt/Component.html#transferFocusUpCycle()), [update](http://docs.google.com/java/awt/Component.html#update(java.awt.Graphics)), [validate](http://docs.google.com/java/awt/Component.html#validate()) |

| **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()), [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)) |

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

### List

public **List**()  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates a new scrolling list. By default, there are four visible lines and multiple selections are not allowed. Note that this is a convenience method for List(0, false). Also note that the number of visible lines in the list cannot be changed after it has been created.

**Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true.**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### List

public **List**(int rows)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates a new scrolling list initialized with the specified number of visible lines. By default, multiple selections are not allowed. Note that this is a convenience method for List(rows, false). Also note that the number of visible rows in the list cannot be changed after it has been created.

**Parameters:**rows - the number of items to show. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true.**Since:** JDK1.1 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### List

public **List**(int rows,  
 boolean multipleMode)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates a new scrolling list initialized to display the specified number of rows. Note that if zero rows are specified, then the list will be created with a default of four rows. Also note that the number of visible rows in the list cannot be changed after it has been created. If the value of multipleMode is true, then the user can select multiple items from the list. If it is false, only one item at a time can be selected.

**Parameters:**rows - the number of items to show.multipleMode - if true, then multiple selections are allowed; otherwise, only one item can be selected at a time. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true.**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

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

### addNotify

public void **addNotify**()

Creates the peer for the list. The peer allows us to modify the list's appearance without changing its functionality.

**Overrides:**[addNotify](http://docs.google.com/java/awt/Component.html#addNotify()) in class [Component](http://docs.google.com/java/awt/Component.html) **See Also:**[Component.isDisplayable()](http://docs.google.com/java/awt/Component.html#isDisplayable()), [Component.removeNotify()](http://docs.google.com/java/awt/Component.html#removeNotify())

### removeNotify

public void **removeNotify**()

Removes the peer for this list. The peer allows us to modify the list's appearance without changing its functionality.

**Overrides:**[removeNotify](http://docs.google.com/java/awt/Component.html#removeNotify()) in class [Component](http://docs.google.com/java/awt/Component.html) **See Also:**[Component.isDisplayable()](http://docs.google.com/java/awt/Component.html#isDisplayable()), [Component.addNotify()](http://docs.google.com/java/awt/Component.html#addNotify())

### getItemCount

public int **getItemCount**()

Gets the number of items in the list.

**Returns:**the number of items in the list**Since:** JDK1.1 **See Also:**[getItem(int)](http://docs.google.com/java/awt/List.html#getItem(int))

### countItems

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public int **countItems**()

**Deprecated.** *As of JDK version 1.1, replaced by getItemCount().*

### getItem

public [String](http://docs.google.com/java/lang/String.html) **getItem**(int index)

Gets the item associated with the specified index.

**Parameters:**index - the position of the item **Returns:**an item that is associated with the specified index**See Also:**[getItemCount()](http://docs.google.com/java/awt/List.html#getItemCount())

### getItems

public [String](http://docs.google.com/java/lang/String.html)[] **getItems**()

Gets the items in the list.

**Returns:**a string array containing items of the list**Since:** JDK1.1 **See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### add

public void **add**([String](http://docs.google.com/java/lang/String.html) item)

Adds the specified item to the end of scrolling list.

**Parameters:**item - the item to be added**Since:** JDK1.1

### addItem

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **addItem**([String](http://docs.google.com/java/lang/String.html) item)

**Deprecated.** *replaced by add(String).*

### add

public void **add**([String](http://docs.google.com/java/lang/String.html) item,  
 int index)

Adds the specified item to the the scrolling list at the position indicated by the index. The index is zero-based. If the value of the index is less than zero, or if the value of the index is greater than or equal to the number of items in the list, then the item is added to the end of the list.

**Parameters:**item - the item to be added; if this parameter is null then the item is treated as an empty string, ""index - the position at which to add the item**Since:** JDK1.1

### addItem

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **addItem**([String](http://docs.google.com/java/lang/String.html) item,  
 int index)

**Deprecated.** *replaced by add(String, int).*

### replaceItem

public void **replaceItem**([String](http://docs.google.com/java/lang/String.html) newValue,  
 int index)

Replaces the item at the specified index in the scrolling list with the new string.

**Parameters:**newValue - a new string to replace an existing itemindex - the position of the item to replace **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if index is out of range

### removeAll

public void **removeAll**()

Removes all items from this list.

**Since:** JDK1.1 **See Also:**[remove(java.lang.String)](http://docs.google.com/java/awt/List.html#remove(java.lang.String)), [delItems(int, int)](http://docs.google.com/java/awt/List.html#delItems(int,%20int))

### clear

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **clear**()

**Deprecated.** *As of JDK version 1.1, replaced by removeAll().*

### remove

public void **remove**([String](http://docs.google.com/java/lang/String.html) item)

Removes the first occurrence of an item from the list. If the specified item is selected, and is the only selected item in the list, the list is set to have no selection.

**Parameters:**item - the item to remove from the list **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the item doesn't exist in the list**Since:** JDK1.1

### remove

public void **remove**(int position)

Removes the item at the specified position from this scrolling list. If the item with the specified position is selected, and is the only selected item in the list, the list is set to have no selection.

**Parameters:**position - the index of the item to delete **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if the position is less than 0 or greater than getItemCount()-1**Since:** JDK1.1 **See Also:**[add(String, int)](http://docs.google.com/java/awt/List.html#add(java.lang.String,%20int))

### delItem

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **delItem**(int position)

**Deprecated.** *replaced by remove(String) and remove(int).*

### getSelectedIndex

public int **getSelectedIndex**()

Gets the index of the selected item on the list,

**Returns:**the index of the selected item; if no item is selected, or if multiple items are selected, -1 is returned.**See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### getSelectedIndexes

public int[] **getSelectedIndexes**()

Gets the selected indexes on the list.

**Returns:**an array of the selected indexes on this scrolling list; if no item is selected, a zero-length array is returned.**See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### getSelectedItem

public [String](http://docs.google.com/java/lang/String.html) **getSelectedItem**()

Gets the selected item on this scrolling list.

**Returns:**the selected item on the list; if no item is selected, or if multiple items are selected, null is returned.**See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### getSelectedItems

public [String](http://docs.google.com/java/lang/String.html)[] **getSelectedItems**()

Gets the selected items on this scrolling list.

**Returns:**an array of the selected items on this scrolling list; if no item is selected, a zero-length array is returned.**See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### getSelectedObjects

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

Gets the selected items on this scrolling list in an array of Objects.

**Specified by:**[getSelectedObjects](http://docs.google.com/java/awt/ItemSelectable.html#getSelectedObjects()) in interface [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html) **Returns:**an array of Objects representing the selected items on this scrolling list; if no item is selected, a zero-length array is returned.**See Also:**[getSelectedItems()](http://docs.google.com/java/awt/List.html#getSelectedItems()), [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html)

### select

public void **select**(int index)

Selects the item at the specified index in the scrolling list.

Note that passing out of range parameters is invalid, and will result in unspecified behavior.

Note that this method should be primarily used to initially select an item in this component. Programmatically calling this method will *not* trigger an ItemEvent. The only way to trigger an ItemEvent is by user interaction.

**Parameters:**index - the position of the item to select**See Also:**[getSelectedItem()](http://docs.google.com/java/awt/List.html#getSelectedItem()), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### deselect

public void **deselect**(int index)

Deselects the item at the specified index.

Note that passing out of range parameters is invalid, and will result in unspecified behavior.

If the item at the specified index is not selected, then the operation is ignored.

**Parameters:**index - the position of the item to deselect**See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [getSelectedItem()](http://docs.google.com/java/awt/List.html#getSelectedItem()), [isIndexSelected(int)](http://docs.google.com/java/awt/List.html#isIndexSelected(int))

### isIndexSelected

public boolean **isIndexSelected**(int index)

Determines if the specified item in this scrolling list is selected.

**Parameters:**index - the item to be checked **Returns:**true if the specified item has been selected; false otherwise**Since:** JDK1.1 **See Also:**[select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int))

### isSelected

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public boolean **isSelected**(int index)

**Deprecated.** *As of JDK version 1.1, replaced by isIndexSelected(int).*

### getRows

public int **getRows**()

Gets the number of visible lines in this list. Note that once the List has been created, this number will never change.

**Returns:**the number of visible lines in this scrolling list

### isMultipleMode

public boolean **isMultipleMode**()

Determines whether this list allows multiple selections.

**Returns:**true if this list allows multiple selections; otherwise, false**Since:** JDK1.1 **See Also:**[setMultipleMode(boolean)](http://docs.google.com/java/awt/List.html#setMultipleMode(boolean))

### allowsMultipleSelections

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public boolean **allowsMultipleSelections**()

**Deprecated.** *As of JDK version 1.1, replaced by isMultipleMode().*

### setMultipleMode

public void **setMultipleMode**(boolean b)

Sets the flag that determines whether this list allows multiple selections. When the selection mode is changed from multiple-selection to single-selection, the selected items change as follows: If a selected item has the location cursor, only that item will remain selected. If no selected item has the location cursor, all items will be deselected.

**Parameters:**b - if true then multiple selections are allowed; otherwise, only one item from the list can be selected at once**Since:** JDK1.1 **See Also:**[isMultipleMode()](http://docs.google.com/java/awt/List.html#isMultipleMode())

### setMultipleSelections

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **setMultipleSelections**(boolean b)

**Deprecated.** *As of JDK version 1.1, replaced by setMultipleMode(boolean).*

### getVisibleIndex

public int **getVisibleIndex**()

Gets the index of the item that was last made visible by the method makeVisible.

**Returns:**the index of the item that was last made visible**See Also:**[makeVisible(int)](http://docs.google.com/java/awt/List.html#makeVisible(int))

### makeVisible

public void **makeVisible**(int index)

Makes the item at the specified index visible.

**Parameters:**index - the position of the item**See Also:**[getVisibleIndex()](http://docs.google.com/java/awt/List.html#getVisibleIndex())

### getPreferredSize

public [Dimension](http://docs.google.com/java/awt/Dimension.html) **getPreferredSize**(int rows)

Gets the preferred dimensions for a list with the specified number of rows.

**Parameters:**rows - number of rows in the list **Returns:**the preferred dimensions for displaying this scrolling list given that the specified number of rows must be visible**Since:** JDK1.1 **See Also:**[Component.getPreferredSize()](http://docs.google.com/java/awt/Component.html#getPreferredSize())

### preferredSize

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public [Dimension](http://docs.google.com/java/awt/Dimension.html) **preferredSize**(int rows)

**Deprecated.** *As of JDK version 1.1, replaced by getPreferredSize(int).*

### getPreferredSize

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

Gets the preferred size of this scrolling list.

**Overrides:**[getPreferredSize](http://docs.google.com/java/awt/Component.html#getPreferredSize()) in class [Component](http://docs.google.com/java/awt/Component.html) **Returns:**the preferred dimensions for displaying this scrolling list**Since:** JDK1.1 **See Also:**[Component.getPreferredSize()](http://docs.google.com/java/awt/Component.html#getPreferredSize())

### preferredSize

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public [Dimension](http://docs.google.com/java/awt/Dimension.html) **preferredSize**()

**Deprecated.** *As of JDK version 1.1, replaced by getPreferredSize().*

**Overrides:**[preferredSize](http://docs.google.com/java/awt/Component.html#preferredSize()) in class [Component](http://docs.google.com/java/awt/Component.html)

### getMinimumSize

public [Dimension](http://docs.google.com/java/awt/Dimension.html) **getMinimumSize**(int rows)

Gets the minumum dimensions for a list with the specified number of rows.

**Parameters:**rows - number of rows in the list **Returns:**the minimum dimensions for displaying this scrolling list given that the specified number of rows must be visible**Since:** JDK1.1 **See Also:**[Component.getMinimumSize()](http://docs.google.com/java/awt/Component.html#getMinimumSize())

### minimumSize

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public [Dimension](http://docs.google.com/java/awt/Dimension.html) **minimumSize**(int rows)

**Deprecated.** *As of JDK version 1.1, replaced by getMinimumSize(int).*

### getMinimumSize

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

Determines the minimum size of this scrolling list.

**Overrides:**[getMinimumSize](http://docs.google.com/java/awt/Component.html#getMinimumSize()) in class [Component](http://docs.google.com/java/awt/Component.html) **Returns:**the minimum dimensions needed to display this scrolling list**Since:** JDK1.1 **See Also:**[Component.getMinimumSize()](http://docs.google.com/java/awt/Component.html#getMinimumSize())

### minimumSize

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public [Dimension](http://docs.google.com/java/awt/Dimension.html) **minimumSize**()

**Deprecated.** *As of JDK version 1.1, replaced by getMinimumSize().*

**Overrides:**[minimumSize](http://docs.google.com/java/awt/Component.html#minimumSize()) in class [Component](http://docs.google.com/java/awt/Component.html)

### addItemListener

public void **addItemListener**([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)

Adds the specified item listener to receive item events from this list. Item events are sent in response to user input, but not in response to calls to select or deselect. If listener l is null, no exception is thrown and no action is performed.

Refer to [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for details on AWT's threading model.

**Specified by:**[addItemListener](http://docs.google.com/java/awt/ItemSelectable.html#addItemListener(java.awt.event.ItemListener)) in interface [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html) **Parameters:**l - the item listener**Since:** JDK1.1 **See Also:**[removeItemListener(java.awt.event.ItemListener)](http://docs.google.com/java/awt/List.html#removeItemListener(java.awt.event.ItemListener)), [getItemListeners()](http://docs.google.com/java/awt/List.html#getItemListeners()), [select(int)](http://docs.google.com/java/awt/List.html#select(int)), [deselect(int)](http://docs.google.com/java/awt/List.html#deselect(int)), [ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html), [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)

### removeItemListener

public void **removeItemListener**([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)

Removes the specified item listener so that it no longer receives item events from this list. If listener l is null, no exception is thrown and no action is performed.

Refer to [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for details on AWT's threading model.

**Specified by:**[removeItemListener](http://docs.google.com/java/awt/ItemSelectable.html#removeItemListener(java.awt.event.ItemListener)) in interface [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html) **Parameters:**l - the item listener**Since:** JDK1.1 **See Also:**[addItemListener(java.awt.event.ItemListener)](http://docs.google.com/java/awt/List.html#addItemListener(java.awt.event.ItemListener)), [getItemListeners()](http://docs.google.com/java/awt/List.html#getItemListeners()), [ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html), [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)

### getItemListeners

public [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)[] **getItemListeners**()

Returns an array of all the item listeners registered on this list.

**Returns:**all of this list's ItemListeners or an empty array if no item listeners are currently registered**Since:** 1.4 **See Also:**[addItemListener(java.awt.event.ItemListener)](http://docs.google.com/java/awt/List.html#addItemListener(java.awt.event.ItemListener)), [removeItemListener(java.awt.event.ItemListener)](http://docs.google.com/java/awt/List.html#removeItemListener(java.awt.event.ItemListener)), [ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html), [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)

### addActionListener

public void **addActionListener**([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)

Adds the specified action listener to receive action events from this list. Action events occur when a user double-clicks on a list item or types Enter when the list has the keyboard focus.

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

Refer to [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for details on AWT's threading model.

**Parameters:**l - the action listener**Since:** JDK1.1 **See Also:**[removeActionListener(java.awt.event.ActionListener)](http://docs.google.com/java/awt/List.html#removeActionListener(java.awt.event.ActionListener)), [getActionListeners()](http://docs.google.com/java/awt/List.html#getActionListeners()), [ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html), [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)

### removeActionListener

public void **removeActionListener**([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)

Removes the specified action listener so that it no longer receives action events from this list. Action events occur when a user double-clicks on a list item. If listener l is null, no exception is thrown and no action is performed.

Refer to [AWT Threading Issues](http://docs.google.com/doc-files/AWTThreadIssues.html#ListenersThreads) for details on AWT's threading model.

**Parameters:**l - the action listener**Since:** JDK1.1 **See Also:**[addActionListener(java.awt.event.ActionListener)](http://docs.google.com/java/awt/List.html#addActionListener(java.awt.event.ActionListener)), [getActionListeners()](http://docs.google.com/java/awt/List.html#getActionListeners()), [ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html), [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)

### getActionListeners

public [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)[] **getActionListeners**()

Returns an array of all the action listeners registered on this list.

**Returns:**all of this list's ActionListeners or an empty array if no action listeners are currently registered**Since:** 1.4 **See Also:**[addActionListener(java.awt.event.ActionListener)](http://docs.google.com/java/awt/List.html#addActionListener(java.awt.event.ActionListener)), [removeActionListener(java.awt.event.ActionListener)](http://docs.google.com/java/awt/List.html#removeActionListener(java.awt.event.ActionListener)), [ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html), [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)

### getListeners

public <T extends [EventListener](http://docs.google.com/java/util/EventListener.html)> T[] **getListeners**([Class](http://docs.google.com/java/lang/Class.html)<T> listenerType)

Returns an array of all the objects currently registered as *Foo*Listeners upon this List. *Foo*Listeners are registered using the add*Foo*Listener method.

You can specify the listenerType argument with a class literal, such as *Foo*Listener.class. For example, you can query a List l for its item listeners with the following code:

ItemListener[] ils = (ItemListener[])(l.getListeners(ItemListener.class));

If no such listeners exist, this method returns an empty array.

**Overrides:**[getListeners](http://docs.google.com/java/awt/Component.html#getListeners(java.lang.Class)) in class [Component](http://docs.google.com/java/awt/Component.html) **Parameters:**listenerType - the type of listeners requested; this parameter should specify an interface that descends from java.util.EventListener **Returns:**an array of all objects registered as *Foo*Listeners on this list, or an empty array if no such listeners have been added **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if listenerType doesn't specify a class or interface that implements java.util.EventListener**Since:** 1.3 **See Also:**[getItemListeners()](http://docs.google.com/java/awt/List.html#getItemListeners())

### processEvent

protected void **processEvent**([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) e)

Processes events on this scrolling list. If an event is an instance of ItemEvent, it invokes the processItemEvent method. Else, if the event is an instance of ActionEvent, it invokes processActionEvent. If the event is not an item event or an action event, it invokes processEvent on the superclass.

Note that if the event parameter is null the behavior is unspecified and may result in an exception.

**Overrides:**[processEvent](http://docs.google.com/java/awt/Component.html#processEvent(java.awt.AWTEvent)) in class [Component](http://docs.google.com/java/awt/Component.html) **Parameters:**e - the event**Since:** JDK1.1 **See Also:**[ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html), [ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html), [processActionEvent(java.awt.event.ActionEvent)](http://docs.google.com/java/awt/List.html#processActionEvent(java.awt.event.ActionEvent)), [processItemEvent(java.awt.event.ItemEvent)](http://docs.google.com/java/awt/List.html#processItemEvent(java.awt.event.ItemEvent))

### processItemEvent

protected void **processItemEvent**([ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html) e)

Processes item events occurring on this list by dispatching them to any registered ItemListener objects.

This method is not called unless item events are enabled for this component. Item events are enabled when one of the following occurs:

* An ItemListener object is registered via addItemListener.
* Item events are enabled via enableEvents.

Note that if the event parameter is null the behavior is unspecified and may result in an exception.

**Parameters:**e - the item event**Since:** JDK1.1 **See Also:**[ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html), [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html), [addItemListener(java.awt.event.ItemListener)](http://docs.google.com/java/awt/List.html#addItemListener(java.awt.event.ItemListener)), [Component.enableEvents(long)](http://docs.google.com/java/awt/Component.html#enableEvents(long))

### processActionEvent

protected void **processActionEvent**([ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html) e)

Processes action events occurring on this component by dispatching them to any registered ActionListener objects.

This method is not called unless action events are enabled for this component. Action events are enabled when one of the following occurs:

* An ActionListener object is registered via addActionListener.
* Action events are enabled via enableEvents.

Note that if the event parameter is null the behavior is unspecified and may result in an exception.

**Parameters:**e - the action event**Since:** JDK1.1 **See Also:**[ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html), [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html), [addActionListener(java.awt.event.ActionListener)](http://docs.google.com/java/awt/List.html#addActionListener(java.awt.event.ActionListener)), [Component.enableEvents(long)](http://docs.google.com/java/awt/Component.html#enableEvents(long))

### paramString

protected [String](http://docs.google.com/java/lang/String.html) **paramString**()

Returns the parameter string representing the state of this scrolling list. This string is useful for debugging.

**Overrides:**[paramString](http://docs.google.com/java/awt/Component.html#paramString()) in class [Component](http://docs.google.com/java/awt/Component.html) **Returns:**the parameter string of this scrolling list

### delItems

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **delItems**(int start,  
 int end)

**Deprecated.** *As of JDK version 1.1, Not for public use in the future. This method is expected to be retained only as a package private method.*

### getAccessibleContext

public [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) **getAccessibleContext**()

Gets the AccessibleContext associated with this List. For lists, the AccessibleContext takes the form of an AccessibleAWTList. A new AccessibleAWTList instance is created, if necessary.

**Specified by:**[getAccessibleContext](http://docs.google.com/javax/accessibility/Accessible.html#getAccessibleContext()) in interface [Accessible](http://docs.google.com/javax/accessibility/Accessible.html)**Overrides:**[getAccessibleContext](http://docs.google.com/java/awt/Component.html#getAccessibleContext()) in class [Component](http://docs.google.com/java/awt/Component.html) **Returns:**an AccessibleAWTList that serves as the AccessibleContext of this List**Since:** 1.3

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/List.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/LinearGradientPaint.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/List.AccessibleAWTList.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/List.html)    [**NO FRAMES**](http://docs.google.com/List.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#3dy6vkm) | [CONSTR](#4d34og8) | [METHOD](#2s8eyo1) | DETAIL: FIELD | [CONSTR](#26in1rg) | [METHOD](#44sinio) |

[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).