| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/BasicMenuItemUI.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuBarUI.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.MouseInputHandler.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/plaf/basic/BasicMenuItemUI.html)    [**NO FRAMES**](http://docs.google.com/BasicMenuItemUI.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#2bn6wsx) | [METHOD](#3as4poj) |

## **javax.swing.plaf.basic**

Class BasicMenuItemUI

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [javax.swing.plaf.ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html)  
 [javax.swing.plaf.ButtonUI](http://docs.google.com/javax/swing/plaf/ButtonUI.html)  
 [javax.swing.plaf.MenuItemUI](http://docs.google.com/javax/swing/plaf/MenuItemUI.html)  
 **javax.swing.plaf.basic.BasicMenuItemUI**

**Direct Known Subclasses:** [BasicCheckBoxMenuItemUI](http://docs.google.com/javax/swing/plaf/basic/BasicCheckBoxMenuItemUI.html), [BasicMenuUI](http://docs.google.com/javax/swing/plaf/basic/BasicMenuUI.html), [BasicRadioButtonMenuItemUI](http://docs.google.com/javax/swing/plaf/basic/BasicRadioButtonMenuItemUI.html)

public class **BasicMenuItemUI**extends [MenuItemUI](http://docs.google.com/javax/swing/plaf/MenuItemUI.html)

BasicMenuItem implementation

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**BasicMenuItemUI.MouseInputHandler**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.MouseInputHandler.html) |

| **Field Summary** | |
| --- | --- |
| protected  [Font](http://docs.google.com/java/awt/Font.html) | [**acceleratorFont**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#acceleratorFont) |
| protected  [Color](http://docs.google.com/java/awt/Color.html) | [**acceleratorForeground**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#acceleratorForeground) |
| protected  [Color](http://docs.google.com/java/awt/Color.html) | [**acceleratorSelectionForeground**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#acceleratorSelectionForeground) |
| protected  [Icon](http://docs.google.com/javax/swing/Icon.html) | [**arrowIcon**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#arrowIcon) |
| protected  [Icon](http://docs.google.com/javax/swing/Icon.html) | [**checkIcon**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#checkIcon) |
| protected  int | [**defaultTextIconGap**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#defaultTextIconGap) |
| protected  [Color](http://docs.google.com/java/awt/Color.html) | [**disabledForeground**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#disabledForeground) |
| protected  [MenuDragMouseListener](http://docs.google.com/javax/swing/event/MenuDragMouseListener.html) | [**menuDragMouseListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#menuDragMouseListener) |
| protected  [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) | [**menuItem**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#menuItem) |
| protected  [MenuKeyListener](http://docs.google.com/javax/swing/event/MenuKeyListener.html) | [**menuKeyListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#menuKeyListener) |
| protected  [MouseInputListener](http://docs.google.com/javax/swing/event/MouseInputListener.html) | [**mouseInputListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#mouseInputListener) |
| protected  boolean | [**oldBorderPainted**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#oldBorderPainted) |
| protected  [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) | [**propertyChangeListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#propertyChangeListener)            PropertyChangeListener returned from createPropertyChangeListener. |
| protected  [Color](http://docs.google.com/java/awt/Color.html) | [**selectionBackground**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#selectionBackground) |
| protected  [Color](http://docs.google.com/java/awt/Color.html) | [**selectionForeground**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#selectionForeground) |

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

| **Method Summary** | |
| --- | --- |
| protected  [MenuDragMouseListener](http://docs.google.com/javax/swing/event/MenuDragMouseListener.html) | [**createMenuDragMouseListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#createMenuDragMouseListener(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c) |
| protected  [MenuKeyListener](http://docs.google.com/javax/swing/event/MenuKeyListener.html) | [**createMenuKeyListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#createMenuKeyListener(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c) |
| protected  [MouseInputListener](http://docs.google.com/javax/swing/event/MouseInputListener.html) | [**createMouseInputListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#createMouseInputListener(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c) |
| protected  [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) | [**createPropertyChangeListener**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#createPropertyChangeListener(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Creates a PropertyChangeListener which will be added to the menu item. |
| static [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) | [**createUI**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#createUI(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c) |
| protected  void | [**doClick**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#doClick(javax.swing.MenuSelectionManager))([MenuSelectionManager](http://docs.google.com/javax/swing/MenuSelectionManager.html) msm)            Call this method when a menu item is to be activated. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getMaximumSize**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#getMaximumSize(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Returns the specified component's maximum size appropriate for the look and feel. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getMinimumSize**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#getMinimumSize(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Returns the specified component's minimum size appropriate for the look and feel. |
| [MenuElement](http://docs.google.com/javax/swing/MenuElement.html)[] | [**getPath**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#getPath())() |
| protected  [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredMenuItemSize**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#getPreferredMenuItemSize(javax.swing.JComponent,%20javax.swing.Icon,%20javax.swing.Icon,%20int))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c, [Icon](http://docs.google.com/javax/swing/Icon.html) checkIcon, [Icon](http://docs.google.com/javax/swing/Icon.html) arrowIcon, int defaultTextIconGap) |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredSize**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#getPreferredSize(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Returns the specified component's preferred size appropriate for the look and feel. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**getPropertyPrefix**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#getPropertyPrefix())() |
| protected  void | [**installComponents**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#installComponents(javax.swing.JMenuItem))([JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem) |
| protected  void | [**installDefaults**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#installDefaults())() |
| protected  void | [**installKeyboardActions**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#installKeyboardActions())() |
| protected  void | [**installListeners**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#installListeners())() |
| void | [**installUI**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#installUI(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Configures the specified component appropriate for the look and feel. |
| void | [**paint**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#paint(java.awt.Graphics,%20javax.swing.JComponent))([Graphics](http://docs.google.com/java/awt/Graphics.html) g, [JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Paints the specified component appropriate for the look and feel. |
| protected  void | [**paintBackground**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#paintBackground(java.awt.Graphics,%20javax.swing.JMenuItem,%20java.awt.Color))([Graphics](http://docs.google.com/java/awt/Graphics.html) g, [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem, [Color](http://docs.google.com/java/awt/Color.html) bgColor)            Draws the background of the menu item. |
| protected  void | [**paintMenuItem**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#paintMenuItem(java.awt.Graphics,%20javax.swing.JComponent,%20javax.swing.Icon,%20javax.swing.Icon,%20java.awt.Color,%20java.awt.Color,%20int))([Graphics](http://docs.google.com/java/awt/Graphics.html) g, [JComponent](http://docs.google.com/javax/swing/JComponent.html) c, [Icon](http://docs.google.com/javax/swing/Icon.html) checkIcon, [Icon](http://docs.google.com/javax/swing/Icon.html) arrowIcon, [Color](http://docs.google.com/java/awt/Color.html) background, [Color](http://docs.google.com/java/awt/Color.html) foreground, int defaultTextIconGap) |
| protected  void | [**paintText**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#paintText(java.awt.Graphics,%20javax.swing.JMenuItem,%20java.awt.Rectangle,%20java.lang.String))([Graphics](http://docs.google.com/java/awt/Graphics.html) g, [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem, [Rectangle](http://docs.google.com/java/awt/Rectangle.html) textRect, [String](http://docs.google.com/java/lang/String.html) text)            Renders the text of the current menu item. |
| protected  void | [**uninstallComponents**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#uninstallComponents(javax.swing.JMenuItem))([JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem) |
| protected  void | [**uninstallDefaults**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#uninstallDefaults())() |
| protected  void | [**uninstallKeyboardActions**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#uninstallKeyboardActions())() |
| protected  void | [**uninstallListeners**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#uninstallListeners())() |
| void | [**uninstallUI**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#uninstallUI(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Reverses configuration which was done on the specified component during installUI. |
| void | [**update**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#update(java.awt.Graphics,%20javax.swing.JComponent))([Graphics](http://docs.google.com/java/awt/Graphics.html) g, [JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            We draw the background in paintMenuItem() so override update (which fills the background of opaque components by default) to just call paint(). |

| **Methods inherited from class javax.swing.plaf.**[**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html) |
| --- |
| [contains](http://docs.google.com/javax/swing/plaf/ComponentUI.html#contains(javax.swing.JComponent,%20int,%20int)), [getAccessibleChild](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getAccessibleChild(javax.swing.JComponent,%20int)), [getAccessibleChildrenCount](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getAccessibleChildrenCount(javax.swing.JComponent)), [getBaseline](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getBaseline(javax.swing.JComponent,%20int,%20int)), [getBaselineResizeBehavior](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getBaselineResizeBehavior(javax.swing.JComponent)) |

| **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** |
| --- |

### menuItem

protected [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) **menuItem**

### selectionBackground

protected [Color](http://docs.google.com/java/awt/Color.html) **selectionBackground**

### selectionForeground

protected [Color](http://docs.google.com/java/awt/Color.html) **selectionForeground**

### disabledForeground

protected [Color](http://docs.google.com/java/awt/Color.html) **disabledForeground**

### acceleratorForeground

protected [Color](http://docs.google.com/java/awt/Color.html) **acceleratorForeground**

### acceleratorSelectionForeground

protected [Color](http://docs.google.com/java/awt/Color.html) **acceleratorSelectionForeground**

### defaultTextIconGap

protected int **defaultTextIconGap**

### acceleratorFont

protected [Font](http://docs.google.com/java/awt/Font.html) **acceleratorFont**

### mouseInputListener

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

### menuDragMouseListener

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

### menuKeyListener

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

### propertyChangeListener

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

PropertyChangeListener returned from createPropertyChangeListener. You should not need to access this field, rather if you want to customize the PropertyChangeListener override createPropertyChangeListener.

**Since:** 1.6 **See Also:**[createPropertyChangeListener(javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.html#createPropertyChangeListener(javax.swing.JComponent))

### arrowIcon

protected [Icon](http://docs.google.com/javax/swing/Icon.html) **arrowIcon**

### checkIcon

protected [Icon](http://docs.google.com/javax/swing/Icon.html) **checkIcon**

### oldBorderPainted

protected boolean **oldBorderPainted**

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

### BasicMenuItemUI

public **BasicMenuItemUI**()

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

### createUI

public static [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **createUI**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

### installUI

public void **installUI**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

**Description copied from class:** [**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#installUI(javax.swing.JComponent)) Configures the specified component appropriate for the look and feel. This method is invoked when the ComponentUI instance is being installed as the UI delegate on the specified component. This method should completely configure the component for the look and feel, including the following:

1. Install any default property values for color, fonts, borders, icons, opacity, etc. on the component. Whenever possible, property values initialized by the client program should *not* be overridden.
2. Install a LayoutManager on the component if necessary.
3. Create/add any required sub-components to the component.
4. Create/install event listeners on the component.
5. Create/install a PropertyChangeListener on the component in order to detect and respond to component property changes appropriately.
6. Install keyboard UI (mnemonics, traversal, etc.) on the component.
7. Initialize any appropriate instance data.

**Overrides:**[installUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html#installUI(javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**c - the component where this UI delegate is being installed**See Also:**[ComponentUI.uninstallUI(javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#uninstallUI(javax.swing.JComponent)), [JComponent.setUI(javax.swing.plaf.ComponentUI)](http://docs.google.com/javax/swing/JComponent.html#setUI(javax.swing.plaf.ComponentUI)), [JComponent.updateUI()](http://docs.google.com/javax/swing/JComponent.html#updateUI())

### installDefaults

protected void **installDefaults**()

### installComponents

protected void **installComponents**([JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem)

**Since:** 1.3

### getPropertyPrefix

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

### installListeners

protected void **installListeners**()

### installKeyboardActions

protected void **installKeyboardActions**()

### uninstallUI

public void **uninstallUI**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

**Description copied from class:** [**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#uninstallUI(javax.swing.JComponent)) Reverses configuration which was done on the specified component during installUI. This method is invoked when this UIComponent instance is being removed as the UI delegate for the specified component. This method should undo the configuration performed in installUI, being careful to leave the JComponent instance in a clean state (no extraneous listeners, look-and-feel-specific property objects, etc.). This should include the following:

1. Remove any UI-set borders from the component.
2. Remove any UI-set layout managers on the component.
3. Remove any UI-added sub-components from the component.
4. Remove any UI-added event/property listeners from the component.
5. Remove any UI-installed keyboard UI from the component.
6. Nullify any allocated instance data objects to allow for GC.

**Overrides:**[uninstallUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html#uninstallUI(javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**c - the component from which this UI delegate is being removed; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple components**See Also:**[ComponentUI.installUI(javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#installUI(javax.swing.JComponent)), [JComponent.updateUI()](http://docs.google.com/javax/swing/JComponent.html#updateUI())

### uninstallDefaults

protected void **uninstallDefaults**()

### uninstallComponents

protected void **uninstallComponents**([JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem)

**Since:** 1.3

### uninstallListeners

protected void **uninstallListeners**()

### uninstallKeyboardActions

protected void **uninstallKeyboardActions**()

### createMouseInputListener

protected [MouseInputListener](http://docs.google.com/javax/swing/event/MouseInputListener.html) **createMouseInputListener**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

### createMenuDragMouseListener

protected [MenuDragMouseListener](http://docs.google.com/javax/swing/event/MenuDragMouseListener.html) **createMenuDragMouseListener**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

### createMenuKeyListener

protected [MenuKeyListener](http://docs.google.com/javax/swing/event/MenuKeyListener.html) **createMenuKeyListener**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

### createPropertyChangeListener

protected [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) **createPropertyChangeListener**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

Creates a PropertyChangeListener which will be added to the menu item. If this method returns null then it will not be added to the menu item.

**Returns:**an instance of a PropertyChangeListener or null**Since:** 1.6

### getMinimumSize

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

**Description copied from class:** [**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getMinimumSize(javax.swing.JComponent)) Returns the specified component's minimum size appropriate for the look and feel. If null is returned, the minimum size will be calculated by the component's layout manager instead (this is the preferred approach for any component with a specific layout manager installed). The default implementation of this method invokes getPreferredSize and returns that value.

**Overrides:**[getMinimumSize](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getMinimumSize(javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**c - the component whose minimum size is being queried; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple components **Returns:**a Dimension object or null**See Also:**[JComponent.getMinimumSize()](http://docs.google.com/javax/swing/JComponent.html#getMinimumSize()), [LayoutManager.minimumLayoutSize(java.awt.Container)](http://docs.google.com/java/awt/LayoutManager.html#minimumLayoutSize(java.awt.Container)), [ComponentUI.getPreferredSize(javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getPreferredSize(javax.swing.JComponent))

### getPreferredSize

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

**Description copied from class:** [**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getPreferredSize(javax.swing.JComponent)) Returns the specified component's preferred size appropriate for the look and feel. If null is returned, the preferred size will be calculated by the component's layout manager instead (this is the preferred approach for any component with a specific layout manager installed). The default implementation of this method returns null.

**Overrides:**[getPreferredSize](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getPreferredSize(javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**c - the component whose preferred size is being queried; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple components**See Also:**[JComponent.getPreferredSize()](http://docs.google.com/javax/swing/JComponent.html#getPreferredSize()), [LayoutManager.preferredLayoutSize(java.awt.Container)](http://docs.google.com/java/awt/LayoutManager.html#preferredLayoutSize(java.awt.Container))

### getMaximumSize

public [Dimension](http://docs.google.com/java/awt/Dimension.html) **getMaximumSize**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

**Description copied from class:** [**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getMaximumSize(javax.swing.JComponent)) Returns the specified component's maximum size appropriate for the look and feel. If null is returned, the maximum size will be calculated by the component's layout manager instead (this is the preferred approach for any component with a specific layout manager installed). The default implementation of this method invokes getPreferredSize and returns that value.

**Overrides:**[getMaximumSize](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getMaximumSize(javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**c - the component whose maximum size is being queried; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple components **Returns:**a Dimension object or null**See Also:**[JComponent.getMaximumSize()](http://docs.google.com/javax/swing/JComponent.html#getMaximumSize()), [LayoutManager2.maximumLayoutSize(java.awt.Container)](http://docs.google.com/java/awt/LayoutManager2.html#maximumLayoutSize(java.awt.Container))

### getPreferredMenuItemSize

protected [Dimension](http://docs.google.com/java/awt/Dimension.html) **getPreferredMenuItemSize**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) checkIcon,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) arrowIcon,  
 int defaultTextIconGap)

### update

public void **update**([Graphics](http://docs.google.com/java/awt/Graphics.html) g,  
 [JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

We draw the background in paintMenuItem() so override update (which fills the background of opaque components by default) to just call paint().

**Overrides:**[update](http://docs.google.com/javax/swing/plaf/ComponentUI.html#update(java.awt.Graphics,%20javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**g - the Graphics context in which to paintc - the component being painted; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple components**See Also:**[ComponentUI.paint(java.awt.Graphics, javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#paint(java.awt.Graphics,%20javax.swing.JComponent)), [JComponent.paintComponent(java.awt.Graphics)](http://docs.google.com/javax/swing/JComponent.html#paintComponent(java.awt.Graphics))

### paint

public void **paint**([Graphics](http://docs.google.com/java/awt/Graphics.html) g,  
 [JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

**Description copied from class:** [**ComponentUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#paint(java.awt.Graphics,%20javax.swing.JComponent)) Paints the specified component appropriate for the look and feel. This method is invoked from the ComponentUI.update method when the specified component is being painted. Subclasses should override this method and use the specified Graphics object to render the content of the component.

**Overrides:**[paint](http://docs.google.com/javax/swing/plaf/ComponentUI.html#paint(java.awt.Graphics,%20javax.swing.JComponent)) in class [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) **Parameters:**g - the Graphics context in which to paintc - the component being painted; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple components**See Also:**[ComponentUI.update(java.awt.Graphics, javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#update(java.awt.Graphics,%20javax.swing.JComponent))

### paintMenuItem

protected void **paintMenuItem**([Graphics](http://docs.google.com/java/awt/Graphics.html) g,  
 [JComponent](http://docs.google.com/javax/swing/JComponent.html) c,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) checkIcon,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) arrowIcon,  
 [Color](http://docs.google.com/java/awt/Color.html) background,  
 [Color](http://docs.google.com/java/awt/Color.html) foreground,  
 int defaultTextIconGap)

### paintBackground

protected void **paintBackground**([Graphics](http://docs.google.com/java/awt/Graphics.html) g,  
 [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem,  
 [Color](http://docs.google.com/java/awt/Color.html) bgColor)

Draws the background of the menu item.

**Parameters:**g - the paint graphicsmenuItem - menu item to be paintedbgColor - selection background color**Since:** 1.4

### paintText

protected void **paintText**([Graphics](http://docs.google.com/java/awt/Graphics.html) g,  
 [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html) menuItem,  
 [Rectangle](http://docs.google.com/java/awt/Rectangle.html) textRect,  
 [String](http://docs.google.com/java/lang/String.html) text)

Renders the text of the current menu item.

**Parameters:**g - graphics contextmenuItem - menu item to rendertextRect - bounding rectangle for rendering the texttext - string to render**Since:** 1.4

### getPath

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

### doClick

protected void **doClick**([MenuSelectionManager](http://docs.google.com/javax/swing/MenuSelectionManager.html) msm)

Call this method when a menu item is to be activated. This method handles some of the details of menu item activation such as clearing the selected path and messaging the JMenuItem's doClick() method.

**Parameters:**msm - A MenuSelectionManager. The visual feedback and internal bookkeeping tasks are delegated to this MenuSelectionManager. If null is passed as this argument, the MenuSelectionManager.defaultManager is used.**Since:** 1.4 **See Also:**[MenuSelectionManager](http://docs.google.com/javax/swing/MenuSelectionManager.html), [AbstractButton.doClick(int)](http://docs.google.com/javax/swing/AbstractButton.html#doClick(int))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/BasicMenuItemUI.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuBarUI.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/plaf/basic/BasicMenuItemUI.MouseInputHandler.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/plaf/basic/BasicMenuItemUI.html)    [**NO FRAMES**](http://docs.google.com/BasicMenuItemUI.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#2bn6wsx) | [METHOD](#3as4poj) |

[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).