| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ComponentUI.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/ComponentInputMapUIResource.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/plaf/DesktopIconUI.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/plaf/ComponentUI.html)    [**NO FRAMES**](http://docs.google.com/ComponentUI.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

## **javax.swing.plaf**

Class ComponentUI

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

**Direct Known Subclasses:** [ButtonUI](http://docs.google.com/javax/swing/plaf/ButtonUI.html), [ColorChooserUI](http://docs.google.com/javax/swing/plaf/ColorChooserUI.html), [ComboBoxUI](http://docs.google.com/javax/swing/plaf/ComboBoxUI.html), [DesktopIconUI](http://docs.google.com/javax/swing/plaf/DesktopIconUI.html), [DesktopPaneUI](http://docs.google.com/javax/swing/plaf/DesktopPaneUI.html), [FileChooserUI](http://docs.google.com/javax/swing/plaf/FileChooserUI.html), [InternalFrameUI](http://docs.google.com/javax/swing/plaf/InternalFrameUI.html), [LabelUI](http://docs.google.com/javax/swing/plaf/LabelUI.html), [ListUI](http://docs.google.com/javax/swing/plaf/ListUI.html), [MenuBarUI](http://docs.google.com/javax/swing/plaf/MenuBarUI.html), [OptionPaneUI](http://docs.google.com/javax/swing/plaf/OptionPaneUI.html), [PanelUI](http://docs.google.com/javax/swing/plaf/PanelUI.html), [PopupMenuUI](http://docs.google.com/javax/swing/plaf/PopupMenuUI.html), [ProgressBarUI](http://docs.google.com/javax/swing/plaf/ProgressBarUI.html), [RootPaneUI](http://docs.google.com/javax/swing/plaf/RootPaneUI.html), [ScrollBarUI](http://docs.google.com/javax/swing/plaf/ScrollBarUI.html), [ScrollPaneUI](http://docs.google.com/javax/swing/plaf/ScrollPaneUI.html), [SeparatorUI](http://docs.google.com/javax/swing/plaf/SeparatorUI.html), [SliderUI](http://docs.google.com/javax/swing/plaf/SliderUI.html), [SpinnerUI](http://docs.google.com/javax/swing/plaf/SpinnerUI.html), [SplitPaneUI](http://docs.google.com/javax/swing/plaf/SplitPaneUI.html), [TabbedPaneUI](http://docs.google.com/javax/swing/plaf/TabbedPaneUI.html), [TableHeaderUI](http://docs.google.com/javax/swing/plaf/TableHeaderUI.html), [TableUI](http://docs.google.com/javax/swing/plaf/TableUI.html), [TextUI](http://docs.google.com/javax/swing/plaf/TextUI.html), [ToolBarUI](http://docs.google.com/javax/swing/plaf/ToolBarUI.html), [ToolTipUI](http://docs.google.com/javax/swing/plaf/ToolTipUI.html), [TreeUI](http://docs.google.com/javax/swing/plaf/TreeUI.html), [ViewportUI](http://docs.google.com/javax/swing/plaf/ViewportUI.html)

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

The base class for all UI delegate objects in the Swing pluggable look and feel architecture. The UI delegate object for a Swing component is responsible for implementing the aspects of the component that depend on the look and feel. The JComponent class invokes methods from this class in order to delegate operations (painting, layout calculations, etc.) that may vary depending on the look and feel installed. **Client programs should not invoke methods on this class directly.**

**See Also:**[JComponent](http://docs.google.com/javax/swing/JComponent.html), [UIManager](http://docs.google.com/javax/swing/UIManager.html)

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

| **Method Summary** | |
| --- | --- |
| boolean | [**contains**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#contains(javax.swing.JComponent,%20int,%20int))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c, int x, int y)            Returns true if the specified *x,y* location is contained within the look and feel's defined shape of the specified component. |
| static [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html) | [**createUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#createUI(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Returns an instance of the UI delegate for the specified component. |
| [Accessible](http://docs.google.com/javax/accessibility/Accessible.html) | [**getAccessibleChild**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getAccessibleChild(javax.swing.JComponent,%20int))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c, int i)            Returns the ith Accessible child of the object. |
| int | [**getAccessibleChildrenCount**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getAccessibleChildrenCount(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Returns the number of accessible children in the object. |
| int | [**getBaseline**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getBaseline(javax.swing.JComponent,%20int,%20int))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c, int width, int height)            Returns the baseline. |
| [Component.BaselineResizeBehavior](http://docs.google.com/java/awt/Component.BaselineResizeBehavior.html) | [**getBaselineResizeBehavior**](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getBaselineResizeBehavior(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Returns an enum indicating how the baseline of he component changes as the size changes. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getMaximumSize**](http://docs.google.com/javax/swing/plaf/ComponentUI.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/ComponentUI.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. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredSize**](http://docs.google.com/javax/swing/plaf/ComponentUI.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. |
| void | [**installUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.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/ComponentUI.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. |
| void | [**uninstallUI**](http://docs.google.com/javax/swing/plaf/ComponentUI.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/ComponentUI.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)            Notifies this UI delegate that it's time to paint the specified component. |

| **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)) |

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

### ComponentUI

public **ComponentUI**()

Sole constructor. (For invocation by subclass constructors, typically implicit.)

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

### installUI

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

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.

**Parameters:**c - the component where this UI delegate is being installed**See Also:**[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())

### uninstallUI

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

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.

**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:**[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())

### paint

public void **paint**([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. 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.

**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:**[update(java.awt.Graphics, javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#update(java.awt.Graphics,%20javax.swing.JComponent))

### update

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

Notifies this UI delegate that it's time to paint the specified component. This method is invoked by JComponent when the specified component is being painted. By default this method will fill the specified component with its background color (if its opaque property is true) and then immediately call paint. In general this method need not be overridden by subclasses; all look-and-feel rendering code should reside in the paint method.

**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:**[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))

### getPreferredSize

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

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.

**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))

### getMinimumSize

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

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.

**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)), [getPreferredSize(javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getPreferredSize(javax.swing.JComponent))

### getMaximumSize

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

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.

**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))

### contains

public boolean **contains**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c,  
 int x,  
 int y)

Returns true if the specified *x,y* location is contained within the look and feel's defined shape of the specified component. x and y are defined to be relative to the coordinate system of the specified component. Although a component's bounds is constrained to a rectangle, this method provides the means for defining a non-rectangular shape within those bounds for the purpose of hit detection.

**Parameters:**c - the component where the *x,y* location is being queried; this argument is often ignored, but might be used if the UI object is stateless and shared by multiple componentsx - the *x* coordinate of the pointy - the *y* coordinate of the point**See Also:**[JComponent.contains(int, int)](http://docs.google.com/javax/swing/JComponent.html#contains(int,%20int)), [Component.contains(int, int)](http://docs.google.com/java/awt/Component.html#contains(int,%20int))

### createUI

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

Returns an instance of the UI delegate for the specified component. Each subclass must provide its own static createUI method that returns an instance of that UI delegate subclass. If the UI delegate subclass is stateless, it may return an instance that is shared by multiple components. If the UI delegate is stateful, then it should return a new instance per component. The default implementation of this method throws an error, as it should never be invoked.

### getBaseline

public int **getBaseline**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c,  
 int width,  
 int height)

Returns the baseline. The baseline is measured from the top of the component. This method is primarily meant for LayoutManagers to align components along their baseline. A return value less than 0 indicates this component does not have a reasonable baseline and that LayoutManagers should not align this component on its baseline.

This method returns -1. Subclasses that have a meaningful baseline should override appropriately.

**Parameters:**c - JComponent baseline is being requested forwidth - the width to get the baseline forheight - the height to get the baseline for **Returns:**baseline or a value < 0 indicating there is no reasonable baseline **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if c is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if width or height is < 0**Since:** 1.6 **See Also:**[JComponent.getBaseline(int,int)](http://docs.google.com/javax/swing/JComponent.html#getBaseline(int,%20int))

### getBaselineResizeBehavior

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

Returns an enum indicating how the baseline of he component changes as the size changes. This method is primarily meant for layout managers and GUI builders.

This method returns BaselineResizeBehavior.OTHER. Subclasses that support a baseline should override appropriately.

**Parameters:**c - JComponent to return baseline resize behavior for **Returns:**an enum indicating how the baseline changes as the component size changes **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if c is null**Since:** 1.6 **See Also:**[JComponent.getBaseline(int, int)](http://docs.google.com/javax/swing/JComponent.html#getBaseline(int,%20int))

### getAccessibleChildrenCount

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

Returns the number of accessible children in the object. If all of the children of this object implement Accessible, this method should return the number of children of this object. UIs might wish to override this if they present areas on the screen that can be viewed as components, but actual components are not used for presenting those areas. Note: As of v1.3, it is recommended that developers call Component.AccessibleAWTComponent.getAccessibleChildrenCount() instead of this method.

**Returns:**the number of accessible children in the object**See Also:**[getAccessibleChild(javax.swing.JComponent, int)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getAccessibleChild(javax.swing.JComponent,%20int))

### getAccessibleChild

public [Accessible](http://docs.google.com/javax/accessibility/Accessible.html) **getAccessibleChild**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c,  
 int i)

Returns the ith Accessible child of the object. UIs might need to override this if they present areas on the screen that can be viewed as components, but actual components are not used for presenting those areas.

Note: As of v1.3, it is recommended that developers call Component.AccessibleAWTComponent.getAccessibleChild() instead of this method.

**Parameters:**i - zero-based index of child **Returns:**the ith Accessible child of the object**See Also:**[getAccessibleChildrenCount(javax.swing.JComponent)](http://docs.google.com/javax/swing/plaf/ComponentUI.html#getAccessibleChildrenCount(javax.swing.JComponent))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ComponentUI.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/ComponentInputMapUIResource.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/plaf/DesktopIconUI.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/plaf/ComponentUI.html)    [**NO FRAMES**](http://docs.google.com/ComponentUI.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

[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).