| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RepaintManager.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/Renderer.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/RootPaneContainer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/RepaintManager.html)    [**NO FRAMES**](http://docs.google.com/RepaintManager.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**

Class RepaintManager

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

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

This class manages repaint requests, allowing the number of repaints to be minimized, for example by collapsing multiple requests into a single repaint for members of a component tree.

As of 1.6 RepaintManager handles repaint requests for Swing's top level components (JApplet, JWindow, JFrame and JDialog). Any calls to repaint on one of these will call into the appropriate addDirtyRegion method.

| **Constructor Summary** | |
| --- | --- |
| [**RepaintManager**](http://docs.google.com/javax/swing/RepaintManager.html#RepaintManager())()            Create a new RepaintManager instance. |

| **Method Summary** | |
| --- | --- |
| void | [**addDirtyRegion**](http://docs.google.com/javax/swing/RepaintManager.html#addDirtyRegion(java.applet.Applet,%20int,%20int,%20int,%20int))([Applet](http://docs.google.com/java/applet/Applet.html) applet, int x, int y, int w, int h)            Adds applet to the list of Components that need to be repainted. |
| void | [**addDirtyRegion**](http://docs.google.com/javax/swing/RepaintManager.html#addDirtyRegion(javax.swing.JComponent,%20int,%20int,%20int,%20int))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c, int x, int y, int w, int h)            Add a component in the list of components that should be refreshed. |
| void | [**addDirtyRegion**](http://docs.google.com/javax/swing/RepaintManager.html#addDirtyRegion(java.awt.Window,%20int,%20int,%20int,%20int))([Window](http://docs.google.com/java/awt/Window.html) window, int x, int y, int w, int h)            Adds window to the list of Components that need to be repainted. |
| void | [**addInvalidComponent**](http://docs.google.com/javax/swing/RepaintManager.html#addInvalidComponent(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) invalidComponent)            Mark the component as in need of layout and queue a runnable for the event dispatching thread that will validate the components first isValidateRoot() ancestor. |
| static [RepaintManager](http://docs.google.com/javax/swing/RepaintManager.html) | [**currentManager**](http://docs.google.com/javax/swing/RepaintManager.html#currentManager(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) c)            Return the RepaintManager for the calling thread given a Component. |
| static [RepaintManager](http://docs.google.com/javax/swing/RepaintManager.html) | [**currentManager**](http://docs.google.com/javax/swing/RepaintManager.html#currentManager(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)            Return the RepaintManager for the calling thread given a JComponent. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getDirtyRegion**](http://docs.google.com/javax/swing/RepaintManager.html#getDirtyRegion(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) aComponent)            Return the current dirty region for a component. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getDoubleBufferMaximumSize**](http://docs.google.com/javax/swing/RepaintManager.html#getDoubleBufferMaximumSize())()            Returns the maximum double buffer size. |
| [Image](http://docs.google.com/java/awt/Image.html) | [**getOffscreenBuffer**](http://docs.google.com/javax/swing/RepaintManager.html#getOffscreenBuffer(java.awt.Component,%20int,%20int))([Component](http://docs.google.com/java/awt/Component.html) c, int proposedWidth, int proposedHeight)            Return the offscreen buffer that should be used as a double buffer with the component c. |
| [Image](http://docs.google.com/java/awt/Image.html) | [**getVolatileOffscreenBuffer**](http://docs.google.com/javax/swing/RepaintManager.html#getVolatileOffscreenBuffer(java.awt.Component,%20int,%20int))([Component](http://docs.google.com/java/awt/Component.html) c, int proposedWidth, int proposedHeight)            Return a volatile offscreen buffer that should be used as a double buffer with the specified component c. |
| boolean | [**isCompletelyDirty**](http://docs.google.com/javax/swing/RepaintManager.html#isCompletelyDirty(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) aComponent)            Convenience method that returns true if **aComponent** will be completely painted during the next paintDirtyRegions(). |
| boolean | [**isDoubleBufferingEnabled**](http://docs.google.com/javax/swing/RepaintManager.html#isDoubleBufferingEnabled())()            Returns true if this RepaintManager is double buffered. |
| void | [**markCompletelyClean**](http://docs.google.com/javax/swing/RepaintManager.html#markCompletelyClean(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) aComponent)            Mark a component completely clean. |
| void | [**markCompletelyDirty**](http://docs.google.com/javax/swing/RepaintManager.html#markCompletelyDirty(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) aComponent)            Mark a component completely dirty. |
| void | [**paintDirtyRegions**](http://docs.google.com/javax/swing/RepaintManager.html#paintDirtyRegions())()            Paint all of the components that have been marked dirty. |
| void | [**removeInvalidComponent**](http://docs.google.com/javax/swing/RepaintManager.html#removeInvalidComponent(javax.swing.JComponent))([JComponent](http://docs.google.com/javax/swing/JComponent.html) component)            Remove a component from the list of invalid components. |
| static void | [**setCurrentManager**](http://docs.google.com/javax/swing/RepaintManager.html#setCurrentManager(javax.swing.RepaintManager))([RepaintManager](http://docs.google.com/javax/swing/RepaintManager.html) aRepaintManager)            Set the RepaintManager that should be used for the calling thread. |
| void | [**setDoubleBufferingEnabled**](http://docs.google.com/javax/swing/RepaintManager.html#setDoubleBufferingEnabled(boolean))(boolean aFlag)            Enables or disables double buffering in this RepaintManager. |
| void | [**setDoubleBufferMaximumSize**](http://docs.google.com/javax/swing/RepaintManager.html#setDoubleBufferMaximumSize(java.awt.Dimension))([Dimension](http://docs.google.com/java/awt/Dimension.html) d)            Set the maximum double buffer size. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/javax/swing/RepaintManager.html#toString())()            Returns a string that displays and identifies this object's properties. |
| void | [**validateInvalidComponents**](http://docs.google.com/javax/swing/RepaintManager.html#validateInvalidComponents())()            Validate all of the components that have been marked invalid. |

| **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** |
| --- |

### RepaintManager

public **RepaintManager**()

Create a new RepaintManager instance. You rarely call this constructor. directly. To get the default RepaintManager, use RepaintManager.currentManager(JComponent) (normally "this").

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

### currentManager

public static [RepaintManager](http://docs.google.com/javax/swing/RepaintManager.html) **currentManager**([Component](http://docs.google.com/java/awt/Component.html) c)

Return the RepaintManager for the calling thread given a Component.

**Parameters:**c - a Component -- unused in the default implementation, but could be used by an overridden version to return a different RepaintManager depending on the Component **Returns:**the RepaintManager object

### currentManager

public static [RepaintManager](http://docs.google.com/javax/swing/RepaintManager.html) **currentManager**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c)

Return the RepaintManager for the calling thread given a JComponent.

Note: This method exists for backward binary compatibility with earlier versions of the Swing library. It simply returns the result returned by [currentManager(Component)](http://docs.google.com/javax/swing/RepaintManager.html#currentManager(java.awt.Component)).

**Parameters:**c - a JComponent -- unused **Returns:**the RepaintManager object

### setCurrentManager

public static void **setCurrentManager**([RepaintManager](http://docs.google.com/javax/swing/RepaintManager.html) aRepaintManager)

Set the RepaintManager that should be used for the calling thread. **aRepaintManager** will become the current RepaintManager for the calling thread's thread group.

**Parameters:**aRepaintManager - the RepaintManager object to use

### addInvalidComponent

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

Mark the component as in need of layout and queue a runnable for the event dispatching thread that will validate the components first isValidateRoot() ancestor.

**See Also:**[JComponent.isValidateRoot()](http://docs.google.com/javax/swing/JComponent.html#isValidateRoot()), [removeInvalidComponent(javax.swing.JComponent)](http://docs.google.com/javax/swing/RepaintManager.html#removeInvalidComponent(javax.swing.JComponent))

### removeInvalidComponent

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

Remove a component from the list of invalid components.

**See Also:**[addInvalidComponent(javax.swing.JComponent)](http://docs.google.com/javax/swing/RepaintManager.html#addInvalidComponent(javax.swing.JComponent))

### addDirtyRegion

public void **addDirtyRegion**([JComponent](http://docs.google.com/javax/swing/JComponent.html) c,  
 int x,  
 int y,  
 int w,  
 int h)

Add a component in the list of components that should be refreshed. If *c* already has a dirty region, the rectangle *(x,y,w,h)* will be unioned with the region that should be redrawn.

**Parameters:**c - Component to repaint, null results in nothing happening.x - X coordinate of the region to repainty - Y coordinate of the region to repaintw - Width of the region to repainth - Height of the region to repaint**See Also:**[JComponent.repaint(long, int, int, int, int)](http://docs.google.com/javax/swing/JComponent.html#repaint(long,%20int,%20int,%20int,%20int))

### addDirtyRegion

public void **addDirtyRegion**([Window](http://docs.google.com/java/awt/Window.html) window,  
 int x,  
 int y,  
 int w,  
 int h)

Adds window to the list of Components that need to be repainted.

**Parameters:**window - Window to repaint, null results in nothing happening.x - X coordinate of the region to repainty - Y coordinate of the region to repaintw - Width of the region to repainth - Height of the region to repaint**Since:** 1.6 **See Also:**[JFrame.repaint(long, int, int, int, int)](http://docs.google.com/javax/swing/JFrame.html#repaint(long,%20int,%20int,%20int,%20int)), [JWindow.repaint(long, int, int, int, int)](http://docs.google.com/javax/swing/JWindow.html#repaint(long,%20int,%20int,%20int,%20int)), [JDialog.repaint(long, int, int, int, int)](http://docs.google.com/javax/swing/JDialog.html#repaint(long,%20int,%20int,%20int,%20int))

### addDirtyRegion

public void **addDirtyRegion**([Applet](http://docs.google.com/java/applet/Applet.html) applet,  
 int x,  
 int y,  
 int w,  
 int h)

Adds applet to the list of Components that need to be repainted.

**Parameters:**applet - Applet to repaint, null results in nothing happening.x - X coordinate of the region to repainty - Y coordinate of the region to repaintw - Width of the region to repainth - Height of the region to repaint**Since:** 1.6 **See Also:**[JApplet.repaint(long, int, int, int, int)](http://docs.google.com/javax/swing/JApplet.html#repaint(long,%20int,%20int,%20int,%20int))

### getDirtyRegion

public [Rectangle](http://docs.google.com/java/awt/Rectangle.html) **getDirtyRegion**([JComponent](http://docs.google.com/javax/swing/JComponent.html) aComponent)

Return the current dirty region for a component. Return an empty rectangle if the component is not dirty.

### markCompletelyDirty

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

Mark a component completely dirty. **aComponent** will be completely painted during the next paintDirtyRegions() call.

### markCompletelyClean

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

Mark a component completely clean. **aComponent** will not get painted during the next paintDirtyRegions() call.

### isCompletelyDirty

public boolean **isCompletelyDirty**([JComponent](http://docs.google.com/javax/swing/JComponent.html) aComponent)

Convenience method that returns true if **aComponent** will be completely painted during the next paintDirtyRegions(). If computing dirty regions is expensive for your component, use this method and avoid computing dirty region if it return true.

### validateInvalidComponents

public void **validateInvalidComponents**()

Validate all of the components that have been marked invalid.

**See Also:**[addInvalidComponent(javax.swing.JComponent)](http://docs.google.com/javax/swing/RepaintManager.html#addInvalidComponent(javax.swing.JComponent))

### paintDirtyRegions

public void **paintDirtyRegions**()

Paint all of the components that have been marked dirty.

**See Also:**[addDirtyRegion(javax.swing.JComponent, int, int, int, int)](http://docs.google.com/javax/swing/RepaintManager.html#addDirtyRegion(javax.swing.JComponent,%20int,%20int,%20int,%20int))

### toString

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

Returns a string that displays and identifies this object's properties.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a String representation of this object

### getOffscreenBuffer

public [Image](http://docs.google.com/java/awt/Image.html) **getOffscreenBuffer**([Component](http://docs.google.com/java/awt/Component.html) c,  
 int proposedWidth,  
 int proposedHeight)

Return the offscreen buffer that should be used as a double buffer with the component c. By default there is a double buffer per RepaintManager. The buffer might be smaller than (proposedWidth,proposedHeight) This happens when the maximum double buffer size as been set for the receiving repaint manager.

### getVolatileOffscreenBuffer

public [Image](http://docs.google.com/java/awt/Image.html) **getVolatileOffscreenBuffer**([Component](http://docs.google.com/java/awt/Component.html) c,  
 int proposedWidth,  
 int proposedHeight)

Return a volatile offscreen buffer that should be used as a double buffer with the specified component c. The image returned will be an instance of VolatileImage, or null if a VolatileImage object could not be instantiated. This buffer might be smaller than (proposedWidth,proposedHeight). This happens when the maximum double buffer size has been set for this repaint manager.

**Since:** 1.4 **See Also:**[VolatileImage](http://docs.google.com/java/awt/image/VolatileImage.html)

### setDoubleBufferMaximumSize

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

Set the maximum double buffer size.

### getDoubleBufferMaximumSize

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

Returns the maximum double buffer size.

**Returns:**a Dimension object representing the maximum size

### setDoubleBufferingEnabled

public void **setDoubleBufferingEnabled**(boolean aFlag)

Enables or disables double buffering in this RepaintManager. CAUTION: The default value for this property is set for optimal paint performance on the given platform and it is not recommended that programs modify this property directly.

**Parameters:**aFlag - true to activate double buffering**See Also:**[isDoubleBufferingEnabled()](http://docs.google.com/javax/swing/RepaintManager.html#isDoubleBufferingEnabled())

### isDoubleBufferingEnabled

public boolean **isDoubleBufferingEnabled**()

Returns true if this RepaintManager is double buffered. The default value for this property may vary from platform to platform. On platforms where native double buffering is supported in the AWT, the default value will be false to avoid unnecessary buffering in Swing. On platforms where native double buffering is not supported, the default value will be true.

**Returns:**true if this object is double buffered

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RepaintManager.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/Renderer.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/RootPaneContainer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/RepaintManager.html)    [**NO FRAMES**](http://docs.google.com/RepaintManager.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).