| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Toolkit.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/TexturePaint.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/Transparency.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/Toolkit.html)    [**NO FRAMES**](http://docs.google.com/Toolkit.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

## **java.awt**

Class Toolkit

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

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

This class is the abstract superclass of all actual implementations of the Abstract Window Toolkit. Subclasses of Toolkit are used to bind the various components to particular native toolkit implementations.

Many GUI operations may be performed asynchronously. This means that if you set the state of a component, and then immediately query the state, the returned value may not yet reflect the requested change. This includes, but is not limited to:

* Scrolling to a specified position.  
  For example, calling ScrollPane.setScrollPosition and then getScrollPosition may return an incorrect value if the original request has not yet been processed.
* Moving the focus from one component to another.  
  For more information, see [Timing Focus Transfers](http://java.sun.com/docs/books/tutorial/uiswing/misc/focus.html#transferTiming), a section in [The Swing Tutorial](http://java.sun.com/docs/books/tutorial/uiswing/).
* Making a top-level container visible.  
  Calling setVisible(true) on a Window, Frame or Dialog may occur asynchronously.
* Setting the size or location of a top-level container.  
  Calls to setSize, setBounds or setLocation on a Window, Frame or Dialog are forwarded to the underlying window management system and may be ignored or modified. See [Window](http://docs.google.com/java/awt/Window.html) for more information.

Most applications should not call any of the methods in this class directly. The methods defined by Toolkit are the "glue" that joins the platform-independent classes in the java.awt package with their counterparts in java.awt.peer. Some methods defined by Toolkit query the native operating system directly.

**Since:** JDK1.0

| **Field Summary** | |
| --- | --- |
| protected  [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Object](http://docs.google.com/java/lang/Object.html)> | [**desktopProperties**](http://docs.google.com/java/awt/Toolkit.html#desktopProperties) |
| protected  [PropertyChangeSupport](http://docs.google.com/java/beans/PropertyChangeSupport.html) | [**desktopPropsSupport**](http://docs.google.com/java/awt/Toolkit.html#desktopPropsSupport) |

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

| **Method Summary** | |
| --- | --- |
| void | [**addAWTEventListener**](http://docs.google.com/java/awt/Toolkit.html#addAWTEventListener(java.awt.event.AWTEventListener,%20long))([AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html) listener, long eventMask)            Adds an AWTEventListener to receive all AWTEvents dispatched system-wide that conform to the given eventMask. |
| void | [**addPropertyChangeListener**](http://docs.google.com/java/awt/Toolkit.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))([String](http://docs.google.com/java/lang/String.html) name, [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) pcl)            Adds the specified property change listener for the named desktop property. |
| abstract  void | [**beep**](http://docs.google.com/java/awt/Toolkit.html#beep())()            Emits an audio beep. |
| abstract  int | [**checkImage**](http://docs.google.com/java/awt/Toolkit.html#checkImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver))([Image](http://docs.google.com/java/awt/Image.html) image, int width, int height, [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html) observer)            Indicates the construction status of a specified image that is being prepared for display. |
| protected abstract  java.awt.peer.ButtonPeer | [**createButton**](http://docs.google.com/java/awt/Toolkit.html#createButton(java.awt.Button))([Button](http://docs.google.com/java/awt/Button.html) target)            Creates this toolkit's implementation of Button using the specified peer interface. |
| protected abstract  java.awt.peer.CanvasPeer | [**createCanvas**](http://docs.google.com/java/awt/Toolkit.html#createCanvas(java.awt.Canvas))([Canvas](http://docs.google.com/java/awt/Canvas.html) target)            Creates this toolkit's implementation of Canvas using the specified peer interface. |
| protected abstract  java.awt.peer.CheckboxPeer | [**createCheckbox**](http://docs.google.com/java/awt/Toolkit.html#createCheckbox(java.awt.Checkbox))([Checkbox](http://docs.google.com/java/awt/Checkbox.html) target)            Creates this toolkit's implementation of Checkbox using the specified peer interface. |
| protected abstract  java.awt.peer.CheckboxMenuItemPeer | [**createCheckboxMenuItem**](http://docs.google.com/java/awt/Toolkit.html#createCheckboxMenuItem(java.awt.CheckboxMenuItem))([CheckboxMenuItem](http://docs.google.com/java/awt/CheckboxMenuItem.html) target)            Creates this toolkit's implementation of CheckboxMenuItem using the specified peer interface. |
| protected abstract  java.awt.peer.ChoicePeer | [**createChoice**](http://docs.google.com/java/awt/Toolkit.html#createChoice(java.awt.Choice))([Choice](http://docs.google.com/java/awt/Choice.html) target)            Creates this toolkit's implementation of Choice using the specified peer interface. |
| protected  java.awt.peer.LightweightPeer | [**createComponent**](http://docs.google.com/java/awt/Toolkit.html#createComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) target)            Creates a peer for a component or container. |
| [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**createCustomCursor**](http://docs.google.com/java/awt/Toolkit.html#createCustomCursor(java.awt.Image,%20java.awt.Point,%20java.lang.String))([Image](http://docs.google.com/java/awt/Image.html) cursor, [Point](http://docs.google.com/java/awt/Point.html) hotSpot, [String](http://docs.google.com/java/lang/String.html) name)            Creates a new custom cursor object. |
| protected abstract  java.awt.peer.DesktopPeer | [**createDesktopPeer**](http://docs.google.com/java/awt/Toolkit.html#createDesktopPeer(java.awt.Desktop))([Desktop](http://docs.google.com/java/awt/Desktop.html) target)            Creates this toolkit's implementation of the Desktop using the specified peer interface. |
| protected abstract  java.awt.peer.DialogPeer | [**createDialog**](http://docs.google.com/java/awt/Toolkit.html#createDialog(java.awt.Dialog))([Dialog](http://docs.google.com/java/awt/Dialog.html) target)            Creates this toolkit's implementation of Dialog using the specified peer interface. |
| | <T extends [DragGestureRecognizer](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html)>  T | | --- | | [**createDragGestureRecognizer**](http://docs.google.com/java/awt/Toolkit.html#createDragGestureRecognizer(java.lang.Class,%20java.awt.dnd.DragSource,%20java.awt.Component,%20int,%20java.awt.dnd.DragGestureListener))([Class](http://docs.google.com/java/lang/Class.html)<T> abstractRecognizerClass, [DragSource](http://docs.google.com/java/awt/dnd/DragSource.html) ds, [Component](http://docs.google.com/java/awt/Component.html) c, int srcActions, [DragGestureListener](http://docs.google.com/java/awt/dnd/DragGestureListener.html) dgl)            Creates a concrete, platform dependent, subclass of the abstract DragGestureRecognizer class requested, and associates it with the DragSource, Component and DragGestureListener specified. |
| abstract  java.awt.dnd.peer.DragSourceContextPeer | [**createDragSourceContextPeer**](http://docs.google.com/java/awt/Toolkit.html#createDragSourceContextPeer(java.awt.dnd.DragGestureEvent))([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) dge)            Creates the peer for a DragSourceContext. |
| protected abstract  java.awt.peer.FileDialogPeer | [**createFileDialog**](http://docs.google.com/java/awt/Toolkit.html#createFileDialog(java.awt.FileDialog))([FileDialog](http://docs.google.com/java/awt/FileDialog.html) target)            Creates this toolkit's implementation of FileDialog using the specified peer interface. |
| protected abstract  java.awt.peer.FramePeer | [**createFrame**](http://docs.google.com/java/awt/Toolkit.html#createFrame(java.awt.Frame))([Frame](http://docs.google.com/java/awt/Frame.html) target)            Creates this toolkit's implementation of Frame using the specified peer interface. |
| [Image](http://docs.google.com/java/awt/Image.html) | [**createImage**](http://docs.google.com/java/awt/Toolkit.html#createImage(byte%5B%5D))(byte[] imagedata)            Creates an image which decodes the image stored in the specified byte array. |
| abstract  [Image](http://docs.google.com/java/awt/Image.html) | [**createImage**](http://docs.google.com/java/awt/Toolkit.html#createImage(byte%5B%5D,%20int,%20int))(byte[] imagedata, int imageoffset, int imagelength)            Creates an image which decodes the image stored in the specified byte array, and at the specified offset and length. |
| abstract  [Image](http://docs.google.com/java/awt/Image.html) | [**createImage**](http://docs.google.com/java/awt/Toolkit.html#createImage(java.awt.image.ImageProducer))([ImageProducer](http://docs.google.com/java/awt/image/ImageProducer.html) producer)            Creates an image with the specified image producer. |
| abstract  [Image](http://docs.google.com/java/awt/Image.html) | [**createImage**](http://docs.google.com/java/awt/Toolkit.html#createImage(java.lang.String))([String](http://docs.google.com/java/lang/String.html) filename)            Returns an image which gets pixel data from the specified file. |
| abstract  [Image](http://docs.google.com/java/awt/Image.html) | [**createImage**](http://docs.google.com/java/awt/Toolkit.html#createImage(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) url)            Returns an image which gets pixel data from the specified URL. |
| protected abstract  java.awt.peer.LabelPeer | [**createLabel**](http://docs.google.com/java/awt/Toolkit.html#createLabel(java.awt.Label))([Label](http://docs.google.com/java/awt/Label.html) target)            Creates this toolkit's implementation of Label using the specified peer interface. |
| protected abstract  java.awt.peer.ListPeer | [**createList**](http://docs.google.com/java/awt/Toolkit.html#createList(java.awt.List))([List](http://docs.google.com/java/awt/List.html) target)            Creates this toolkit's implementation of List using the specified peer interface. |
| protected abstract  java.awt.peer.MenuPeer | [**createMenu**](http://docs.google.com/java/awt/Toolkit.html#createMenu(java.awt.Menu))([Menu](http://docs.google.com/java/awt/Menu.html) target)            Creates this toolkit's implementation of Menu using the specified peer interface. |
| protected abstract  java.awt.peer.MenuBarPeer | [**createMenuBar**](http://docs.google.com/java/awt/Toolkit.html#createMenuBar(java.awt.MenuBar))([MenuBar](http://docs.google.com/java/awt/MenuBar.html) target)            Creates this toolkit's implementation of MenuBar using the specified peer interface. |
| protected abstract  java.awt.peer.MenuItemPeer | [**createMenuItem**](http://docs.google.com/java/awt/Toolkit.html#createMenuItem(java.awt.MenuItem))([MenuItem](http://docs.google.com/java/awt/MenuItem.html) target)            Creates this toolkit's implementation of MenuItem using the specified peer interface. |
| protected abstract  java.awt.peer.PanelPeer | [**createPanel**](http://docs.google.com/java/awt/Toolkit.html#createPanel(java.awt.Panel))([Panel](http://docs.google.com/java/awt/Panel.html) target)            Creates this toolkit's implementation of Panel using the specified peer interface. |
| protected abstract  java.awt.peer.PopupMenuPeer | [**createPopupMenu**](http://docs.google.com/java/awt/Toolkit.html#createPopupMenu(java.awt.PopupMenu))([PopupMenu](http://docs.google.com/java/awt/PopupMenu.html) target)            Creates this toolkit's implementation of PopupMenu using the specified peer interface. |
| protected abstract  java.awt.peer.ScrollbarPeer | [**createScrollbar**](http://docs.google.com/java/awt/Toolkit.html#createScrollbar(java.awt.Scrollbar))([Scrollbar](http://docs.google.com/java/awt/Scrollbar.html) target)            Creates this toolkit's implementation of Scrollbar using the specified peer interface. |
| protected abstract  java.awt.peer.ScrollPanePeer | [**createScrollPane**](http://docs.google.com/java/awt/Toolkit.html#createScrollPane(java.awt.ScrollPane))([ScrollPane](http://docs.google.com/java/awt/ScrollPane.html) target)            Creates this toolkit's implementation of ScrollPane using the specified peer interface. |
| protected abstract  java.awt.peer.TextAreaPeer | [**createTextArea**](http://docs.google.com/java/awt/Toolkit.html#createTextArea(java.awt.TextArea))([TextArea](http://docs.google.com/java/awt/TextArea.html) target)            Creates this toolkit's implementation of TextArea using the specified peer interface. |
| protected abstract  java.awt.peer.TextFieldPeer | [**createTextField**](http://docs.google.com/java/awt/Toolkit.html#createTextField(java.awt.TextField))([TextField](http://docs.google.com/java/awt/TextField.html) target)            Creates this toolkit's implementation of TextField using the specified peer interface. |
| protected abstract  java.awt.peer.WindowPeer | [**createWindow**](http://docs.google.com/java/awt/Toolkit.html#createWindow(java.awt.Window))([Window](http://docs.google.com/java/awt/Window.html) target)            Creates this toolkit's implementation of Window using the specified peer interface. |
| [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html)[] | [**getAWTEventListeners**](http://docs.google.com/java/awt/Toolkit.html#getAWTEventListeners())()            Returns an array of all the AWTEventListeners registered on this toolkit. |
| [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html)[] | [**getAWTEventListeners**](http://docs.google.com/java/awt/Toolkit.html#getAWTEventListeners(long))(long eventMask)            Returns an array of all the AWTEventListeners registered on this toolkit which listen to all of the event types specified in the eventMask argument. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getBestCursorSize**](http://docs.google.com/java/awt/Toolkit.html#getBestCursorSize(int,%20int))(int preferredWidth, int preferredHeight)            Returns the supported cursor dimension which is closest to the desired sizes. |
| abstract  [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) | [**getColorModel**](http://docs.google.com/java/awt/Toolkit.html#getColorModel())()            Determines the color model of this toolkit's screen. |
| static [Toolkit](http://docs.google.com/java/awt/Toolkit.html) | [**getDefaultToolkit**](http://docs.google.com/java/awt/Toolkit.html#getDefaultToolkit())()            Gets the default toolkit. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getDesktopProperty**](http://docs.google.com/java/awt/Toolkit.html#getDesktopProperty(java.lang.String))([String](http://docs.google.com/java/lang/String.html) propertyName)            Obtains a value for the specified desktop property. |
| abstract  [String](http://docs.google.com/java/lang/String.html)[] | [**getFontList**](http://docs.google.com/java/awt/Toolkit.html#getFontList())()  **Deprecated.** *see* [*GraphicsEnvironment.getAvailableFontFamilyNames()*](http://docs.google.com/java/awt/GraphicsEnvironment.html#getAvailableFontFamilyNames()) |
| abstract  [FontMetrics](http://docs.google.com/java/awt/FontMetrics.html) | [**getFontMetrics**](http://docs.google.com/java/awt/Toolkit.html#getFontMetrics(java.awt.Font))([Font](http://docs.google.com/java/awt/Font.html) font)  **Deprecated.** *As of JDK version 1.2, replaced by the Font method getLineMetrics.* |
| protected abstract  java.awt.peer.FontPeer | [**getFontPeer**](http://docs.google.com/java/awt/Toolkit.html#getFontPeer(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) name, int style)  **Deprecated.** *see java.awt.GraphicsEnvironment#getAllFonts* |
| abstract  [Image](http://docs.google.com/java/awt/Image.html) | [**getImage**](http://docs.google.com/java/awt/Toolkit.html#getImage(java.lang.String))([String](http://docs.google.com/java/lang/String.html) filename)            Returns an image which gets pixel data from the specified file, whose format can be either GIF, JPEG or PNG. |
| abstract  [Image](http://docs.google.com/java/awt/Image.html) | [**getImage**](http://docs.google.com/java/awt/Toolkit.html#getImage(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) url)            Returns an image which gets pixel data from the specified URL. |
| boolean | [**getLockingKeyState**](http://docs.google.com/java/awt/Toolkit.html#getLockingKeyState(int))(int keyCode)            Returns whether the given locking key on the keyboard is currently in its "on" state. |
| int | [**getMaximumCursorColors**](http://docs.google.com/java/awt/Toolkit.html#getMaximumCursorColors())()            Returns the maximum number of colors the Toolkit supports in a custom cursor palette. |
| int | [**getMenuShortcutKeyMask**](http://docs.google.com/java/awt/Toolkit.html#getMenuShortcutKeyMask())()            Determines which modifier key is the appropriate accelerator key for menu shortcuts. |
| protected  java.awt.peer.MouseInfoPeer | [**getMouseInfoPeer**](http://docs.google.com/java/awt/Toolkit.html#getMouseInfoPeer())()            Obtains this toolkit's implementation of helper class for MouseInfo operations. |
| protected static [Container](http://docs.google.com/java/awt/Container.html) | [**getNativeContainer**](http://docs.google.com/java/awt/Toolkit.html#getNativeContainer(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) c)            Give native peers the ability to query the native container given a native component (eg the direct parent may be lightweight). |
| [PrintJob](http://docs.google.com/java/awt/PrintJob.html) | [**getPrintJob**](http://docs.google.com/java/awt/Toolkit.html#getPrintJob(java.awt.Frame,%20java.lang.String,%20java.awt.JobAttributes,%20java.awt.PageAttributes))([Frame](http://docs.google.com/java/awt/Frame.html) frame, [String](http://docs.google.com/java/lang/String.html) jobtitle, [JobAttributes](http://docs.google.com/java/awt/JobAttributes.html) jobAttributes, [PageAttributes](http://docs.google.com/java/awt/PageAttributes.html) pageAttributes)            Gets a PrintJob object which is the result of initiating a print operation on the toolkit's platform. |
| abstract  [PrintJob](http://docs.google.com/java/awt/PrintJob.html) | [**getPrintJob**](http://docs.google.com/java/awt/Toolkit.html#getPrintJob(java.awt.Frame,%20java.lang.String,%20java.util.Properties))([Frame](http://docs.google.com/java/awt/Frame.html) frame, [String](http://docs.google.com/java/lang/String.html) jobtitle, [Properties](http://docs.google.com/java/util/Properties.html) props)            Gets a PrintJob object which is the result of initiating a print operation on the toolkit's platform. |
| static [String](http://docs.google.com/java/lang/String.html) | [**getProperty**](http://docs.google.com/java/awt/Toolkit.html#getProperty(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) key, [String](http://docs.google.com/java/lang/String.html) defaultValue)            Gets a property with the specified key and default. |
| [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html)[] | [**getPropertyChangeListeners**](http://docs.google.com/java/awt/Toolkit.html#getPropertyChangeListeners())()            Returns an array of all the property change listeners registered on this toolkit. |
| [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html)[] | [**getPropertyChangeListeners**](http://docs.google.com/java/awt/Toolkit.html#getPropertyChangeListeners(java.lang.String))([String](http://docs.google.com/java/lang/String.html) propertyName)            Returns an array of all the PropertyChangeListeners associated with the named property. |
| [Insets](http://docs.google.com/java/awt/Insets.html) | [**getScreenInsets**](http://docs.google.com/java/awt/Toolkit.html#getScreenInsets(java.awt.GraphicsConfiguration))([GraphicsConfiguration](http://docs.google.com/java/awt/GraphicsConfiguration.html) gc)            Gets the insets of the screen. |
| abstract  int | [**getScreenResolution**](http://docs.google.com/java/awt/Toolkit.html#getScreenResolution())()            Returns the screen resolution in dots-per-inch. |
| abstract  [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getScreenSize**](http://docs.google.com/java/awt/Toolkit.html#getScreenSize())()            Gets the size of the screen. |
| abstract  [Clipboard](http://docs.google.com/java/awt/datatransfer/Clipboard.html) | [**getSystemClipboard**](http://docs.google.com/java/awt/Toolkit.html#getSystemClipboard())()            Gets the singleton instance of the system Clipboard which interfaces with clipboard facilities provided by the native platform. |
| [EventQueue](http://docs.google.com/java/awt/EventQueue.html) | [**getSystemEventQueue**](http://docs.google.com/java/awt/Toolkit.html#getSystemEventQueue())()            Get the application's or applet's EventQueue instance. |
| protected abstract  [EventQueue](http://docs.google.com/java/awt/EventQueue.html) | [**getSystemEventQueueImpl**](http://docs.google.com/java/awt/Toolkit.html#getSystemEventQueueImpl())()            Gets the application's or applet's EventQueue instance, without checking access. |
| [Clipboard](http://docs.google.com/java/awt/datatransfer/Clipboard.html) | [**getSystemSelection**](http://docs.google.com/java/awt/Toolkit.html#getSystemSelection())()            Gets the singleton instance of the system selection as a Clipboard object. |
| protected  void | [**initializeDesktopProperties**](http://docs.google.com/java/awt/Toolkit.html#initializeDesktopProperties())()            initializeDesktopProperties |
| boolean | [**isAlwaysOnTopSupported**](http://docs.google.com/java/awt/Toolkit.html#isAlwaysOnTopSupported())()            Returns whether the always-on-top mode is supported by this toolkit. |
| boolean | [**isDynamicLayoutActive**](http://docs.google.com/java/awt/Toolkit.html#isDynamicLayoutActive())()            Returns whether dynamic layout of Containers on resize is currently active (both set programmatically, and supported by the underlying operating system and/or window manager). |
| protected  boolean | [**isDynamicLayoutSet**](http://docs.google.com/java/awt/Toolkit.html#isDynamicLayoutSet())()            Returns whether the layout of Containers is validated dynamically during resizing, or statically, after resizing is complete. |
| boolean | [**isFrameStateSupported**](http://docs.google.com/java/awt/Toolkit.html#isFrameStateSupported(int))(int state)            Returns whether Toolkit supports this state for Frames. |
| abstract  boolean | [**isModalExclusionTypeSupported**](http://docs.google.com/java/awt/Toolkit.html#isModalExclusionTypeSupported(java.awt.Dialog.ModalExclusionType))([Dialog.ModalExclusionType](http://docs.google.com/java/awt/Dialog.ModalExclusionType.html) modalExclusionType)            Returns whether the given modal exclusion type is supported by this toolkit. |
| abstract  boolean | [**isModalityTypeSupported**](http://docs.google.com/java/awt/Toolkit.html#isModalityTypeSupported(java.awt.Dialog.ModalityType))([Dialog.ModalityType](http://docs.google.com/java/awt/Dialog.ModalityType.html) modalityType)            Returns whether the given modality type is supported by this toolkit. |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**lazilyLoadDesktopProperty**](http://docs.google.com/java/awt/Toolkit.html#lazilyLoadDesktopProperty(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            an opportunity to lazily evaluate desktop property values. |
| protected  void | [**loadSystemColors**](http://docs.google.com/java/awt/Toolkit.html#loadSystemColors(int%5B%5D))(int[] systemColors)            Fills in the integer array that is supplied as an argument with the current system color values. |
| abstract  [Map](http://docs.google.com/java/util/Map.html)<[TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html),?> | [**mapInputMethodHighlight**](http://docs.google.com/java/awt/Toolkit.html#mapInputMethodHighlight(java.awt.im.InputMethodHighlight))([InputMethodHighlight](http://docs.google.com/java/awt/im/InputMethodHighlight.html) highlight)            Returns a map of visual attributes for the abstract level description of the given input method highlight, or null if no mapping is found. |
| abstract  boolean | [**prepareImage**](http://docs.google.com/java/awt/Toolkit.html#prepareImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver))([Image](http://docs.google.com/java/awt/Image.html) image, int width, int height, [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html) observer)            Prepares an image for rendering. |
| void | [**removeAWTEventListener**](http://docs.google.com/java/awt/Toolkit.html#removeAWTEventListener(java.awt.event.AWTEventListener))([AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html) listener)            Removes an AWTEventListener from receiving dispatched AWTEvents. |
| void | [**removePropertyChangeListener**](http://docs.google.com/java/awt/Toolkit.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener))([String](http://docs.google.com/java/lang/String.html) name, [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) pcl)            Removes the specified property change listener for the named desktop property. |
| protected  void | [**setDesktopProperty**](http://docs.google.com/java/awt/Toolkit.html#setDesktopProperty(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) name, [Object](http://docs.google.com/java/lang/Object.html) newValue)            Sets the named desktop property to the specified value and fires a property change event to notify any listeners that the value has changed. |
| void | [**setDynamicLayout**](http://docs.google.com/java/awt/Toolkit.html#setDynamicLayout(boolean))(boolean dynamic)            Controls whether the layout of Containers is validated dynamically during resizing, or statically, after resizing is complete. |
| void | [**setLockingKeyState**](http://docs.google.com/java/awt/Toolkit.html#setLockingKeyState(int,%20boolean))(int keyCode, boolean on)            Sets the state of the given locking key on the keyboard. |
| abstract  void | [**sync**](http://docs.google.com/java/awt/Toolkit.html#sync())()            Synchronizes this toolkit's graphics state. |

| **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** |
| --- |

### desktopProperties

protected final [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Object](http://docs.google.com/java/lang/Object.html)> **desktopProperties**

### desktopPropsSupport

protected final [PropertyChangeSupport](http://docs.google.com/java/beans/PropertyChangeSupport.html) **desktopPropsSupport**

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

### Toolkit

public **Toolkit**()

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

### createDesktopPeer

protected abstract java.awt.peer.DesktopPeer **createDesktopPeer**([Desktop](http://docs.google.com/java/awt/Desktop.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of the Desktop using the specified peer interface.

**Parameters:**target - the desktop to be implemented **Returns:**this toolkit's implementation of the Desktop **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.6 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless()), [Desktop](http://docs.google.com/java/awt/Desktop.html), DesktopPeer

### createButton

protected abstract java.awt.peer.ButtonPeer **createButton**([Button](http://docs.google.com/java/awt/Button.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Button using the specified peer interface.

**Parameters:**target - the button to be implemented. **Returns:**this toolkit's implementation of Button. **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()), [Button](http://docs.google.com/java/awt/Button.html), ButtonPeer

### createTextField

protected abstract java.awt.peer.TextFieldPeer **createTextField**([TextField](http://docs.google.com/java/awt/TextField.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of TextField using the specified peer interface.

**Parameters:**target - the text field to be implemented. **Returns:**this toolkit's implementation of TextField. **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()), [TextField](http://docs.google.com/java/awt/TextField.html), TextFieldPeer

### createLabel

protected abstract java.awt.peer.LabelPeer **createLabel**([Label](http://docs.google.com/java/awt/Label.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Label using the specified peer interface.

**Parameters:**target - the label to be implemented. **Returns:**this toolkit's implementation of Label. **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()), [Label](http://docs.google.com/java/awt/Label.html), LabelPeer

### createList

protected abstract java.awt.peer.ListPeer **createList**([List](http://docs.google.com/java/awt/List.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of List using the specified peer interface.

**Parameters:**target - the list to be implemented. **Returns:**this toolkit's implementation of List. **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](http://docs.google.com/java/awt/List.html), ListPeer

### createCheckbox

protected abstract java.awt.peer.CheckboxPeer **createCheckbox**([Checkbox](http://docs.google.com/java/awt/Checkbox.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Checkbox using the specified peer interface.

**Parameters:**target - the check box to be implemented. **Returns:**this toolkit's implementation of Checkbox. **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()), [Checkbox](http://docs.google.com/java/awt/Checkbox.html), CheckboxPeer

### createScrollbar

protected abstract java.awt.peer.ScrollbarPeer **createScrollbar**([Scrollbar](http://docs.google.com/java/awt/Scrollbar.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Scrollbar using the specified peer interface.

**Parameters:**target - the scroll bar to be implemented. **Returns:**this toolkit's implementation of Scrollbar. **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()), [Scrollbar](http://docs.google.com/java/awt/Scrollbar.html), ScrollbarPeer

### createScrollPane

protected abstract java.awt.peer.ScrollPanePeer **createScrollPane**([ScrollPane](http://docs.google.com/java/awt/ScrollPane.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of ScrollPane using the specified peer interface.

**Parameters:**target - the scroll pane to be implemented. **Returns:**this toolkit's implementation of ScrollPane. **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()), [ScrollPane](http://docs.google.com/java/awt/ScrollPane.html), ScrollPanePeer

### createTextArea

protected abstract java.awt.peer.TextAreaPeer **createTextArea**([TextArea](http://docs.google.com/java/awt/TextArea.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of TextArea using the specified peer interface.

**Parameters:**target - the text area to be implemented. **Returns:**this toolkit's implementation of TextArea. **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()), [TextArea](http://docs.google.com/java/awt/TextArea.html), TextAreaPeer

### createChoice

protected abstract java.awt.peer.ChoicePeer **createChoice**([Choice](http://docs.google.com/java/awt/Choice.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Choice using the specified peer interface.

**Parameters:**target - the choice to be implemented. **Returns:**this toolkit's implementation of Choice. **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()), [Choice](http://docs.google.com/java/awt/Choice.html), ChoicePeer

### createFrame

protected abstract java.awt.peer.FramePeer **createFrame**([Frame](http://docs.google.com/java/awt/Frame.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Frame using the specified peer interface.

**Parameters:**target - the frame to be implemented. **Returns:**this toolkit's implementation of Frame. **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()), [Frame](http://docs.google.com/java/awt/Frame.html), FramePeer

### createCanvas

protected abstract java.awt.peer.CanvasPeer **createCanvas**([Canvas](http://docs.google.com/java/awt/Canvas.html) target)

Creates this toolkit's implementation of Canvas using the specified peer interface.

**Parameters:**target - the canvas to be implemented. **Returns:**this toolkit's implementation of Canvas.**See Also:**[Canvas](http://docs.google.com/java/awt/Canvas.html), CanvasPeer

### createPanel

protected abstract java.awt.peer.PanelPeer **createPanel**([Panel](http://docs.google.com/java/awt/Panel.html) target)

Creates this toolkit's implementation of Panel using the specified peer interface.

**Parameters:**target - the panel to be implemented. **Returns:**this toolkit's implementation of Panel.**See Also:**[Panel](http://docs.google.com/java/awt/Panel.html), PanelPeer

### createWindow

protected abstract java.awt.peer.WindowPeer **createWindow**([Window](http://docs.google.com/java/awt/Window.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Window using the specified peer interface.

**Parameters:**target - the window to be implemented. **Returns:**this toolkit's implementation of Window. **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()), [Window](http://docs.google.com/java/awt/Window.html), WindowPeer

### createDialog

protected abstract java.awt.peer.DialogPeer **createDialog**([Dialog](http://docs.google.com/java/awt/Dialog.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Dialog using the specified peer interface.

**Parameters:**target - the dialog to be implemented. **Returns:**this toolkit's implementation of Dialog. **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()), [Dialog](http://docs.google.com/java/awt/Dialog.html), DialogPeer

### createMenuBar

protected abstract java.awt.peer.MenuBarPeer **createMenuBar**([MenuBar](http://docs.google.com/java/awt/MenuBar.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of MenuBar using the specified peer interface.

**Parameters:**target - the menu bar to be implemented. **Returns:**this toolkit's implementation of MenuBar. **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()), [MenuBar](http://docs.google.com/java/awt/MenuBar.html), MenuBarPeer

### createMenu

protected abstract java.awt.peer.MenuPeer **createMenu**([Menu](http://docs.google.com/java/awt/Menu.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of Menu using the specified peer interface.

**Parameters:**target - the menu to be implemented. **Returns:**this toolkit's implementation of Menu. **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()), [Menu](http://docs.google.com/java/awt/Menu.html), MenuPeer

### createPopupMenu

protected abstract java.awt.peer.PopupMenuPeer **createPopupMenu**([PopupMenu](http://docs.google.com/java/awt/PopupMenu.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of PopupMenu using the specified peer interface.

**Parameters:**target - the popup menu to be implemented. **Returns:**this toolkit's implementation of PopupMenu. **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()), [PopupMenu](http://docs.google.com/java/awt/PopupMenu.html), PopupMenuPeer

### createMenuItem

protected abstract java.awt.peer.MenuItemPeer **createMenuItem**([MenuItem](http://docs.google.com/java/awt/MenuItem.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of MenuItem using the specified peer interface.

**Parameters:**target - the menu item to be implemented. **Returns:**this toolkit's implementation of MenuItem. **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()), [MenuItem](http://docs.google.com/java/awt/MenuItem.html), MenuItemPeer

### createFileDialog

protected abstract java.awt.peer.FileDialogPeer **createFileDialog**([FileDialog](http://docs.google.com/java/awt/FileDialog.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of FileDialog using the specified peer interface.

**Parameters:**target - the file dialog to be implemented. **Returns:**this toolkit's implementation of FileDialog. **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()), [FileDialog](http://docs.google.com/java/awt/FileDialog.html), FileDialogPeer

### createCheckboxMenuItem

protected abstract java.awt.peer.CheckboxMenuItemPeer **createCheckboxMenuItem**([CheckboxMenuItem](http://docs.google.com/java/awt/CheckboxMenuItem.html) target)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates this toolkit's implementation of CheckboxMenuItem using the specified peer interface.

**Parameters:**target - the checkbox menu item to be implemented. **Returns:**this toolkit's implementation of CheckboxMenuItem. **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()), [CheckboxMenuItem](http://docs.google.com/java/awt/CheckboxMenuItem.html), CheckboxMenuItemPeer

### getMouseInfoPeer

protected java.awt.peer.MouseInfoPeer **getMouseInfoPeer**()

Obtains this toolkit's implementation of helper class for MouseInfo operations.

**Returns:**this toolkit's implementation of helper for MouseInfo **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this operation is not implemented**Since:** 1.5 **See Also:**MouseInfoPeer, [MouseInfo](http://docs.google.com/java/awt/MouseInfo.html)

### createComponent

protected java.awt.peer.LightweightPeer **createComponent**([Component](http://docs.google.com/java/awt/Component.html) target)

Creates a peer for a component or container. This peer is windowless and allows the Component and Container classes to be extended directly to create windowless components that are defined entirely in java.

**Parameters:**target - The Component to be created.

### getFontPeer

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
protected abstract java.awt.peer.FontPeer **getFontPeer**([String](http://docs.google.com/java/lang/String.html) name,  
 int style)

**Deprecated.** *see java.awt.GraphicsEnvironment#getAllFonts*

Creates this toolkit's implementation of Font using the specified peer interface.

**Parameters:**name - the font to be implementedstyle - the style of the font, such as PLAIN, BOLD, ITALIC, or a combination **Returns:**this toolkit's implementation of Font**See Also:**[Font](http://docs.google.com/java/awt/Font.html), FontPeer, [GraphicsEnvironment.getAllFonts()](http://docs.google.com/java/awt/GraphicsEnvironment.html#getAllFonts())

### loadSystemColors

protected void **loadSystemColors**(int[] systemColors)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Fills in the integer array that is supplied as an argument with the current system color values.

**Parameters:**systemColors - an integer array. **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())

### setDynamicLayout

public void **setDynamicLayout**(boolean dynamic)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Controls whether the layout of Containers is validated dynamically during resizing, or statically, after resizing is complete. Note that this feature is not supported on all platforms, and conversely, that this feature cannot be turned off on some platforms. On platforms where dynamic layout during resize is not supported (or is always supported), setting this property has no effect. Note that this feature can be set or unset as a property of the operating system or window manager on some platforms. On such platforms, the dynamic resize property must be set at the operating system or window manager level before this method can take effect. This method does not change the underlying operating system or window manager support or settings. The OS/WM support can be queried using getDesktopProperty("awt.dynamicLayoutSupported").

**Parameters:**dynamic - If true, Containers should re-layout their components as the Container is being resized. If false, the layout will be validated after resizing is finished. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.4 **See Also:**[isDynamicLayoutSet()](http://docs.google.com/java/awt/Toolkit.html#isDynamicLayoutSet()), [isDynamicLayoutActive()](http://docs.google.com/java/awt/Toolkit.html#isDynamicLayoutActive()), [getDesktopProperty(String propertyName)](http://docs.google.com/java/awt/Toolkit.html#getDesktopProperty(java.lang.String)), [GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### isDynamicLayoutSet

protected boolean **isDynamicLayoutSet**()  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Returns whether the layout of Containers is validated dynamically during resizing, or statically, after resizing is complete. Note: this method returns the value that was set programmatically; it does not reflect support at the level of the operating system or window manager for dynamic layout on resizing, or the current operating system or window manager settings. The OS/WM support can be queried using getDesktopProperty("awt.dynamicLayoutSupported").

**Returns:**true if validation of Containers is done dynamically, false if validation is done after resizing is finished. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.4 **See Also:**[setDynamicLayout(boolean dynamic)](http://docs.google.com/java/awt/Toolkit.html#setDynamicLayout(boolean)), [isDynamicLayoutActive()](http://docs.google.com/java/awt/Toolkit.html#isDynamicLayoutActive()), [getDesktopProperty(String propertyName)](http://docs.google.com/java/awt/Toolkit.html#getDesktopProperty(java.lang.String)), [GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### isDynamicLayoutActive

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

Returns whether dynamic layout of Containers on resize is currently active (both set programmatically, and supported by the underlying operating system and/or window manager). The OS/WM support can be queried using getDesktopProperty("awt.dynamicLayoutSupported").

**Returns:**true if dynamic layout of Containers on resize is currently active, false otherwise. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.4 **See Also:**[setDynamicLayout(boolean dynamic)](http://docs.google.com/java/awt/Toolkit.html#setDynamicLayout(boolean)), [isDynamicLayoutSet()](http://docs.google.com/java/awt/Toolkit.html#isDynamicLayoutSet()), [getDesktopProperty(String propertyName)](http://docs.google.com/java/awt/Toolkit.html#getDesktopProperty(java.lang.String)), [GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getScreenSize

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

Gets the size of the screen. On systems with multiple displays, the primary display is used. Multi-screen aware display dimensions are available from GraphicsConfiguration and GraphicsDevice.

**Returns:**the size of this toolkit's screen, in pixels. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**See Also:**[GraphicsConfiguration.getBounds()](http://docs.google.com/java/awt/GraphicsConfiguration.html#getBounds()), [GraphicsDevice.getDisplayMode()](http://docs.google.com/java/awt/GraphicsDevice.html#getDisplayMode()), [GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getScreenResolution

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

Returns the screen resolution in dots-per-inch.

**Returns:**this toolkit's screen resolution, in dots-per-inch. **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())

### getScreenInsets

public [Insets](http://docs.google.com/java/awt/Insets.html) **getScreenInsets**([GraphicsConfiguration](http://docs.google.com/java/awt/GraphicsConfiguration.html) gc)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Gets the insets of the screen.

**Parameters:**gc - a GraphicsConfiguration **Returns:**the insets of this toolkit's screen, in pixels. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.4 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getColorModel

public abstract [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html) **getColorModel**()  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Determines the color model of this toolkit's screen.

ColorModel is an abstract class that encapsulates the ability to translate between the pixel values of an image and its red, green, blue, and alpha components.

This toolkit method is called by the getColorModel method of the Component class.

**Returns:**the color model of this toolkit's screen. **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()), [ColorModel](http://docs.google.com/java/awt/image/ColorModel.html), [Component.getColorModel()](http://docs.google.com/java/awt/Component.html#getColorModel())

### getFontList

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public abstract [String](http://docs.google.com/java/lang/String.html)[] **getFontList**()

**Deprecated.** *see* [*GraphicsEnvironment.getAvailableFontFamilyNames()*](http://docs.google.com/java/awt/GraphicsEnvironment.html#getAvailableFontFamilyNames())

Returns the names of the available fonts in this toolkit.

For 1.1, the following font names are deprecated (the replacement name follows):

* TimesRoman (use Serif)
* Helvetica (use SansSerif)
* Courier (use Monospaced)

The ZapfDingbats fontname is also deprecated in 1.1 but the characters are defined in Unicode starting at 0x2700, and as of 1.1 Java supports those characters.

**Returns:**the names of the available fonts in this toolkit.**See Also:**[GraphicsEnvironment.getAvailableFontFamilyNames()](http://docs.google.com/java/awt/GraphicsEnvironment.html#getAvailableFontFamilyNames())

### getFontMetrics

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public abstract [FontMetrics](http://docs.google.com/java/awt/FontMetrics.html) **getFontMetrics**([Font](http://docs.google.com/java/awt/Font.html) font)

**Deprecated.** *As of JDK version 1.2, replaced by the Font method getLineMetrics.*

Gets the screen device metrics for rendering of the font.

**Parameters:**font - a font **Returns:**the screen metrics of the specified font in this toolkit**See Also:**[LineMetrics](http://docs.google.com/java/awt/font/LineMetrics.html), [Font.getLineMetrics(java.lang.String, java.awt.font.FontRenderContext)](http://docs.google.com/java/awt/Font.html#getLineMetrics(java.lang.String,%20java.awt.font.FontRenderContext)), [GraphicsEnvironment.getScreenDevices()](http://docs.google.com/java/awt/GraphicsEnvironment.html#getScreenDevices())

### sync

public abstract void **sync**()

Synchronizes this toolkit's graphics state. Some window systems may do buffering of graphics events.

This method ensures that the display is up-to-date. It is useful for animation.

### getDefaultToolkit

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

Gets the default toolkit.

If a system property named "java.awt.headless" is set to true then the headless implementation of Toolkit is used.

If there is no "java.awt.headless" or it is set to false and there is a system property named "awt.toolkit", that property is treated as the name of a class that is a subclass of Toolkit; otherwise the default platform-specific implementation of Toolkit is used.

Also loads additional classes into the VM, using the property 'assistive\_technologies' specified in the Sun reference implementation by a line in the 'accessibility.properties' file. The form is "assistive\_technologies=..." where the "..." is a comma-separated list of assistive technology classes to load. Each class is loaded in the order given and a single instance of each is created using Class.forName(class).newInstance(). This is done just after the AWT toolkit is created. All errors are handled via an AWTError exception.

**Returns:**the default toolkit. **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if a toolkit could not be found, or if one could not be accessed or instantiated.

### getImage

public abstract [Image](http://docs.google.com/java/awt/Image.html) **getImage**([String](http://docs.google.com/java/lang/String.html) filename)

Returns an image which gets pixel data from the specified file, whose format can be either GIF, JPEG or PNG. The underlying toolkit attempts to resolve multiple requests with the same filename to the same returned Image.

Since the mechanism required to facilitate this sharing of Image objects may continue to hold onto images that are no longer in use for an indefinite period of time, developers are encouraged to implement their own caching of images by using the [createImage](http://docs.google.com/java/awt/Toolkit.html#createImage(java.lang.String)) variant wherever available. If the image data contained in the specified file changes, the Image object returned from this method may still contain stale information which was loaded from the file after a prior call. Previously loaded image data can be manually discarded by calling the [flush](http://docs.google.com/java/awt/Image.html#flush()) method on the returned Image.

This method first checks if there is a security manager installed. If so, the method calls the security manager's checkRead method with the file specified to ensure that the access to the image is allowed.

**Parameters:**filename - the name of a file containing pixel data in a recognized file format. **Returns:**an image which gets its pixel data from the specified file. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkRead method doesn't allow the operation.**See Also:**[createImage(java.lang.String)](http://docs.google.com/java/awt/Toolkit.html#createImage(java.lang.String))

### getImage

public abstract [Image](http://docs.google.com/java/awt/Image.html) **getImage**([URL](http://docs.google.com/java/net/URL.html) url)

Returns an image which gets pixel data from the specified URL. The pixel data referenced by the specified URL must be in one of the following formats: GIF, JPEG or PNG. The underlying toolkit attempts to resolve multiple requests with the same URL to the same returned Image.

Since the mechanism required to facilitate this sharing of Image objects may continue to hold onto images that are no longer in use for an indefinite period of time, developers are encouraged to implement their own caching of images by using the [createImage](http://docs.google.com/java/awt/Toolkit.html#createImage(java.net.URL)) variant wherever available. If the image data stored at the specified URL changes, the Image object returned from this method may still contain stale information which was fetched from the URL after a prior call. Previously loaded image data can be manually discarded by calling the [flush](http://docs.google.com/java/awt/Image.html#flush()) method on the returned Image.

This method first checks if there is a security manager installed. If so, the method calls the security manager's checkPermission method with the url.openConnection().getPermission() permission to ensure that the access to the image is allowed. For compatibility with pre-1.2 security managers, if the access is denied with FilePermission or SocketPermission, the method throws the SecurityException if the corresponding 1.1-style SecurityManager.checkXXX method also denies permission.

**Parameters:**url - the URL to use in fetching the pixel data. **Returns:**an image which gets its pixel data from the specified URL. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow the operation.**See Also:**[createImage(java.net.URL)](http://docs.google.com/java/awt/Toolkit.html#createImage(java.net.URL))

### createImage

public abstract [Image](http://docs.google.com/java/awt/Image.html) **createImage**([String](http://docs.google.com/java/lang/String.html) filename)

Returns an image which gets pixel data from the specified file. The returned Image is a new object which will not be shared with any other caller of this method or its getImage variant.

This method first checks if there is a security manager installed. If so, the method calls the security manager's checkRead method with the specified file to ensure that the image creation is allowed.

**Parameters:**filename - the name of a file containing pixel data in a recognized file format. **Returns:**an image which gets its pixel data from the specified file. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkRead method doesn't allow the operation.**See Also:**[getImage(java.lang.String)](http://docs.google.com/java/awt/Toolkit.html#getImage(java.lang.String))

### createImage

public abstract [Image](http://docs.google.com/java/awt/Image.html) **createImage**([URL](http://docs.google.com/java/net/URL.html) url)

Returns an image which gets pixel data from the specified URL. The returned Image is a new object which will not be shared with any other caller of this method or its getImage variant.

This method first checks if there is a security manager installed. If so, the method calls the security manager's checkPermission method with the url.openConnection().getPermission() permission to ensure that the image creation is allowed. For compatibility with pre-1.2 security managers, if the access is denied with FilePermission or SocketPermission, the method throws SecurityException if the corresponding 1.1-style SecurityManager.checkXXX method also denies permission.

**Parameters:**url - the URL to use in fetching the pixel data. **Returns:**an image which gets its pixel data from the specified URL. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow the operation.**See Also:**[getImage(java.net.URL)](http://docs.google.com/java/awt/Toolkit.html#getImage(java.net.URL))

### prepareImage

public abstract boolean **prepareImage**([Image](http://docs.google.com/java/awt/Image.html) image,  
 int width,  
 int height,  
 [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html) observer)

Prepares an image for rendering.

If the values of the width and height arguments are both -1, this method prepares the image for rendering on the default screen; otherwise, this method prepares an image for rendering on the default screen at the specified width and height.

The image data is downloaded asynchronously in another thread, and an appropriately scaled screen representation of the image is generated.

This method is called by components prepareImage methods.

Information on the flags returned by this method can be found with the definition of the ImageObserver interface.

**Parameters:**image - the image for which to prepare a screen representation.width - the width of the desired screen representation, or -1.height - the height of the desired screen representation, or -1.observer - the ImageObserver object to be notified as the image is being prepared. **Returns:**true if the image has already been fully prepared; false otherwise.**See Also:**[Component.prepareImage(java.awt.Image, java.awt.image.ImageObserver)](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20java.awt.image.ImageObserver)), [Component.prepareImage(java.awt.Image, int, int, java.awt.image.ImageObserver)](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html)

### checkImage

public abstract int **checkImage**([Image](http://docs.google.com/java/awt/Image.html) image,  
 int width,  
 int height,  
 [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html) observer)

Indicates the construction status of a specified image that is being prepared for display.

If the values of the width and height arguments are both -1, this method returns the construction status of a screen representation of the specified image in this toolkit. Otherwise, this method returns the construction status of a scaled representation of the image at the specified width and height.

This method does not cause the image to begin loading. An application must call prepareImage to force the loading of an image.

This method is called by the component's checkImage methods.

Information on the flags returned by this method can be found with the definition of the ImageObserver interface.

**Parameters:**image - the image whose status is being checked.width - the width of the scaled version whose status is being checked, or -1.height - the height of the scaled version whose status is being checked, or -1.observer - the ImageObserver object to be notified as the image is being prepared. **Returns:**the bitwise inclusive **OR** of the ImageObserver flags for the image data that is currently available.**See Also:**[prepareImage(java.awt.Image, int, int, java.awt.image.ImageObserver)](http://docs.google.com/java/awt/Toolkit.html#prepareImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [Component.checkImage(java.awt.Image, java.awt.image.ImageObserver)](http://docs.google.com/java/awt/Component.html#checkImage(java.awt.Image,%20java.awt.image.ImageObserver)), [Component.checkImage(java.awt.Image, int, int, java.awt.image.ImageObserver)](http://docs.google.com/java/awt/Component.html#checkImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html)

### createImage

public abstract [Image](http://docs.google.com/java/awt/Image.html) **createImage**([ImageProducer](http://docs.google.com/java/awt/image/ImageProducer.html) producer)

Creates an image with the specified image producer.

**Parameters:**producer - the image producer to be used. **Returns:**an image with the specified image producer.**See Also:**[Image](http://docs.google.com/java/awt/Image.html), [ImageProducer](http://docs.google.com/java/awt/image/ImageProducer.html), [Component.createImage(java.awt.image.ImageProducer)](http://docs.google.com/java/awt/Component.html#createImage(java.awt.image.ImageProducer))

### createImage

public [Image](http://docs.google.com/java/awt/Image.html) **createImage**(byte[] imagedata)

Creates an image which decodes the image stored in the specified byte array.

The data must be in some image format, such as GIF or JPEG, that is supported by this toolkit.

**Parameters:**imagedata - an array of bytes, representing image data in a supported image format. **Returns:**an image.**Since:** JDK1.1

### createImage

public abstract [Image](http://docs.google.com/java/awt/Image.html) **createImage**(byte[] imagedata,  
 int imageoffset,  
 int imagelength)

Creates an image which decodes the image stored in the specified byte array, and at the specified offset and length. The data must be in some image format, such as GIF or JPEG, that is supported by this toolkit.

**Parameters:**imagedata - an array of bytes, representing image data in a supported image format.imageoffset - the offset of the beginning of the data in the array.imagelength - the length of the data in the array. **Returns:**an image.**Since:** JDK1.1

### getPrintJob

public abstract [PrintJob](http://docs.google.com/java/awt/PrintJob.html) **getPrintJob**([Frame](http://docs.google.com/java/awt/Frame.html) frame,  
 [String](http://docs.google.com/java/lang/String.html) jobtitle,  
 [Properties](http://docs.google.com/java/util/Properties.html) props)

Gets a PrintJob object which is the result of initiating a print operation on the toolkit's platform.

Each actual implementation of this method should first check if there is a security manager installed. If there is, the method should call the security manager's checkPrintJobAccess method to ensure initiation of a print operation is allowed. If the default implementation of checkPrintJobAccess is used (that is, that method is not overriden), then this results in a call to the security manager's checkPermission method with a RuntimePermission("queuePrintJob") permission.

**Parameters:**frame - the parent of the print dialog. May not be null.jobtitle - the title of the PrintJob. A null title is equivalent to "".props - a Properties object containing zero or more properties. Properties are not standardized and are not consistent across implementations. Because of this, PrintJobs which require job and page control should use the version of this function which takes JobAttributes and PageAttributes objects. This object may be updated to reflect the user's job choices on exit. May be null. **Returns:**a PrintJob object, or null if the user cancelled the print job. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if frame is null. This exception is always thrown when GraphicsEnvironment.isHeadless() returns true. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this thread is not allowed to initiate a print job request**Since:** JDK1.1 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless()), [PrintJob](http://docs.google.com/java/awt/PrintJob.html), [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)

### getPrintJob

public [PrintJob](http://docs.google.com/java/awt/PrintJob.html) **getPrintJob**([Frame](http://docs.google.com/java/awt/Frame.html) frame,  
 [String](http://docs.google.com/java/lang/String.html) jobtitle,  
 [JobAttributes](http://docs.google.com/java/awt/JobAttributes.html) jobAttributes,  
 [PageAttributes](http://docs.google.com/java/awt/PageAttributes.html) pageAttributes)

Gets a PrintJob object which is the result of initiating a print operation on the toolkit's platform.

Each actual implementation of this method should first check if there is a security manager installed. If there is, the method should call the security manager's checkPrintJobAccess method to ensure initiation of a print operation is allowed. If the default implementation of checkPrintJobAccess is used (that is, that method is not overriden), then this results in a call to the security manager's checkPermission method with a RuntimePermission("queuePrintJob") permission.

**Parameters:**frame - the parent of the print dialog. May be null if and only if jobAttributes is not null and jobAttributes.getDialog() returns JobAttributes.DialogType.NONE or JobAttributes.DialogType.COMMON.jobtitle - the title of the PrintJob. A null title is equivalent to "".jobAttributes - a set of job attributes which will control the PrintJob. The attributes will be updated to reflect the user's choices as outlined in the JobAttributes documentation. May be null.pageAttributes - a set of page attributes which will control the PrintJob. The attributes will be applied to every page in the job. The attributes will be updated to reflect the user's choices as outlined in the PageAttributes documentation. May be null. **Returns:**a PrintJob object, or null if the user cancelled the print job. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if frame is null and either jobAttributes is null or jobAttributes.getDialog() returns JobAttributes.DialogType.NATIVE. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if pageAttributes specifies differing cross feed and feed resolutions. Also if this thread has access to the file system and jobAttributes specifies print to file, and the specified destination file exists but is a directory rather than a regular file, does not exist but cannot be created, or cannot be opened for any other reason. However in the case of print to file, if a dialog is also requested to be displayed then the user will be given an opportunity to select a file and proceed with printing. The dialog will ensure that the selected output file is valid before returning from this method.

This exception is always thrown when GraphicsEnvironment.isHeadless() returns true.

[SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if this thread is not allowed to initiate a print job request, or if jobAttributes specifies print to file, and this thread is not allowed to access the file system**Since:** 1.3 **See Also:**[PrintJob](http://docs.google.com/java/awt/PrintJob.html), [GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless()), [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html), [JobAttributes](http://docs.google.com/java/awt/JobAttributes.html), [PageAttributes](http://docs.google.com/java/awt/PageAttributes.html)

### beep

public abstract void **beep**()

Emits an audio beep.

**Since:** JDK1.1

### getSystemClipboard

public abstract [Clipboard](http://docs.google.com/java/awt/datatransfer/Clipboard.html) **getSystemClipboard**()  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Gets the singleton instance of the system Clipboard which interfaces with clipboard facilities provided by the native platform. This clipboard enables data transfer between Java programs and native applications which use native clipboard facilities.

In addition to any and all formats specified in the flavormap.properties file, or other file specified by the AWT.DnD.flavorMapFileURL Toolkit property, text returned by the system Clipboard's getTransferData() method is available in the following flavors:

* DataFlavor.stringFlavor
* DataFlavor.plainTextFlavor (**deprecated**)

As with java.awt.datatransfer.StringSelection, if the requested flavor is DataFlavor.plainTextFlavor, or an equivalent flavor, a Reader is returned. **Note:** The behavior of the system Clipboard's getTransferData() method for DataFlavor.plainTextFlavor, and equivalent DataFlavors, is inconsistent with the definition of DataFlavor.plainTextFlavor . Because of this, support for DataFlavor.plainTextFlavor, and equivalent flavors, is **deprecated**.

Each actual implementation of this method should first check if there is a security manager installed. If there is, the method should call the security manager's checkSystemClipboardAccess method to ensure it's ok to to access the system clipboard. If the default implementation of checkSystemClipboardAccess is used (that is, that method is not overriden), then this results in a call to the security manager's checkPermission method with an AWTPermission("accessClipboard") permission.

**Returns:**the system Clipboard **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()), [Clipboard](http://docs.google.com/java/awt/datatransfer/Clipboard.html), [StringSelection](http://docs.google.com/java/awt/datatransfer/StringSelection.html), [DataFlavor.stringFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#stringFlavor), [DataFlavor.plainTextFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#plainTextFlavor), [Reader](http://docs.google.com/java/io/Reader.html), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html)

### getSystemSelection

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

Gets the singleton instance of the system selection as a Clipboard object. This allows an application to read and modify the current, system-wide selection.

An application is responsible for updating the system selection whenever the user selects text, using either the mouse or the keyboard. Typically, this is implemented by installing a FocusListener on all Components which support text selection, and, between FOCUS\_GAINED and FOCUS\_LOST events delivered to that Component, updating the system selection Clipboard when the selection changes inside the Component. Properly updating the system selection ensures that a Java application will interact correctly with native applications and other Java applications running simultaneously on the system. Note that java.awt.TextComponent and javax.swing.text.JTextComponent already adhere to this policy. When using these classes, and their subclasses, developers need not write any additional code.

Some platforms do not support a system selection Clipboard. On those platforms, this method will return null. In such a case, an application is absolved from its responsibility to update the system selection Clipboard as described above.

Each actual implementation of this method should first check if there is a SecurityManager installed. If there is, the method should call the SecurityManager's checkSystemClipboardAccess method to ensure that client code has access the system selection. If the default implementation of checkSystemClipboardAccess is used (that is, if the method is not overridden), then this results in a call to the SecurityManager's checkPermission method with an AWTPermission("accessClipboard") permission.

**Returns:**the system selection as a Clipboard, or null if the native platform does not support a system selection Clipboard **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.4 **See Also:**[Clipboard](http://docs.google.com/java/awt/datatransfer/Clipboard.html), [FocusListener](http://docs.google.com/java/awt/event/FocusListener.html), [FocusEvent.FOCUS\_GAINED](http://docs.google.com/java/awt/event/FocusEvent.html#FOCUS_GAINED), [FocusEvent.FOCUS\_LOST](http://docs.google.com/java/awt/event/FocusEvent.html#FOCUS_LOST), [TextComponent](http://docs.google.com/java/awt/TextComponent.html), [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html), [GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getMenuShortcutKeyMask

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

Determines which modifier key is the appropriate accelerator key for menu shortcuts.

Menu shortcuts, which are embodied in the MenuShortcut class, are handled by the MenuBar class.

By default, this method returns Event.CTRL\_MASK. Toolkit implementations should override this method if the **Control** key isn't the correct key for accelerators.

**Returns:**the modifier mask on the Event class that is used for menu shortcuts on this toolkit. **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()), [MenuBar](http://docs.google.com/java/awt/MenuBar.html), [MenuShortcut](http://docs.google.com/java/awt/MenuShortcut.html)

### getLockingKeyState

public boolean **getLockingKeyState**(int keyCode)  
 throws [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html)

Returns whether the given locking key on the keyboard is currently in its "on" state. Valid key codes are [VK\_CAPS\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_CAPS_LOCK), [VK\_NUM\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_NUM_LOCK), [VK\_SCROLL\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_SCROLL_LOCK), and [VK\_KANA\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_KANA_LOCK).

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if keyCode is not one of the valid key codes [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if the host system doesn't allow getting the state of this key programmatically, or if the keyboard doesn't have this key [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.3 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### setLockingKeyState

public void **setLockingKeyState**(int keyCode,  
 boolean on)  
 throws [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html)

Sets the state of the given locking key on the keyboard. Valid key codes are [VK\_CAPS\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_CAPS_LOCK), [VK\_NUM\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_NUM_LOCK), [VK\_SCROLL\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_SCROLL_LOCK), and [VK\_KANA\_LOCK](http://docs.google.com/java/awt/event/KeyEvent.html#VK_KANA_LOCK).

Depending on the platform, setting the state of a locking key may involve event processing and therefore may not be immediately observable through getLockingKeyState.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if keyCode is not one of the valid key codes [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if the host system doesn't allow setting the state of this key programmatically, or if the keyboard doesn't have this key [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.3 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getNativeContainer

protected static [Container](http://docs.google.com/java/awt/Container.html) **getNativeContainer**([Component](http://docs.google.com/java/awt/Component.html) c)

Give native peers the ability to query the native container given a native component (eg the direct parent may be lightweight).

### createCustomCursor

public [Cursor](http://docs.google.com/java/awt/Cursor.html) **createCustomCursor**([Image](http://docs.google.com/java/awt/Image.html) cursor,  
 [Point](http://docs.google.com/java/awt/Point.html) hotSpot,  
 [String](http://docs.google.com/java/lang/String.html) name)  
 throws [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html),  
 [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates a new custom cursor object. If the image to display is invalid, the cursor will be hidden (made completely transparent), and the hotspot will be set to (0, 0).

Note that multi-frame images are invalid and may cause this method to hang.

**Parameters:**cursor - the image to display when the cursor is activedhotSpot - the X and Y of the large cursor's hot spot; the hotSpot values must be less than the Dimension returned by getBestCursorSizename - a localized description of the cursor, for Java Accessibility use **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the hotSpot values are outside the bounds of the cursor [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.2 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getBestCursorSize

public [Dimension](http://docs.google.com/java/awt/Dimension.html) **getBestCursorSize**(int preferredWidth,  
 int preferredHeight)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Returns the supported cursor dimension which is closest to the desired sizes. Systems which only support a single cursor size will return that size regardless of the desired sizes. Systems which don't support custom cursors will return a dimension of 0, 0.

Note: if an image is used whose dimensions don't match a supported size (as returned by this method), the Toolkit implementation will attempt to resize the image to a supported size. Since converting low-resolution images is difficult, no guarantees are made as to the quality of a cursor image which isn't a supported size. It is therefore recommended that this method be called and an appropriate image used so no image conversion is made.

**Parameters:**preferredWidth - the preferred cursor width the component would like to use.preferredHeight - the preferred cursor height the component would like to use. **Returns:**the closest matching supported cursor size, or a dimension of 0,0 if the Toolkit implementation doesn't support custom cursors. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.2 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getMaximumCursorColors

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

Returns the maximum number of colors the Toolkit supports in a custom cursor palette.

Note: if an image is used which has more colors in its palette than the supported maximum, the Toolkit implementation will attempt to flatten the palette to the maximum. Since converting low-resolution images is difficult, no guarantees are made as to the quality of a cursor image which has more colors than the system supports. It is therefore recommended that this method be called and an appropriate image used so no image conversion is made.

**Returns:**the maximum number of colors, or zero if custom cursors are not supported by this Toolkit implementation. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**Since:** 1.2 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### isFrameStateSupported

public boolean **isFrameStateSupported**(int state)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Returns whether Toolkit supports this state for Frames. This method tells whether the *UI concept* of, say, maximization or iconification is supported. It will always return false for "compound" states like Frame.ICONIFIED|Frame.MAXIMIZED\_VERT. In other words, the rule of thumb is that only queries with a single frame state constant as an argument are meaningful.

**Parameters:**state - one of named frame state constants. **Returns:**true is this frame state is supported by this Toolkit implementation, false otherwise. **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true.**Since:** 1.4 **See Also:**[Frame.setExtendedState(int)](http://docs.google.com/java/awt/Frame.html#setExtendedState(int))

### getProperty

public static [String](http://docs.google.com/java/lang/String.html) **getProperty**([String](http://docs.google.com/java/lang/String.html) key,  
 [String](http://docs.google.com/java/lang/String.html) defaultValue)

Gets a property with the specified key and default. This method returns defaultValue if the property is not found.

### getSystemEventQueue

public final [EventQueue](http://docs.google.com/java/awt/EventQueue.html) **getSystemEventQueue**()

Get the application's or applet's EventQueue instance. Depending on the Toolkit implementation, different EventQueues may be returned for different applets. Applets should therefore not assume that the EventQueue instance returned by this method will be shared by other applets or the system.

First, if there is a security manager, its checkAwtEventQueueAccess method is called. If the default implementation of checkAwtEventQueueAccess is used (that is, that method is not overriden), then this results in a call to the security manager's checkPermission method with an AWTPermission("accessEventQueue") permission.

**Returns:**the EventQueue object **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its [SecurityManager.checkAwtEventQueueAccess()](http://docs.google.com/java/lang/SecurityManager.html#checkAwtEventQueueAccess()) method denies access to the EventQueue**See Also:**[AWTPermission](http://docs.google.com/java/awt/AWTPermission.html)

### getSystemEventQueueImpl

protected abstract [EventQueue](http://docs.google.com/java/awt/EventQueue.html) **getSystemEventQueueImpl**()

Gets the application's or applet's EventQueue instance, without checking access. For security reasons, this can only be called from a Toolkit subclass.

**Returns:**the EventQueue object

### createDragSourceContextPeer

public abstract java.awt.dnd.peer.DragSourceContextPeer **createDragSourceContextPeer**([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) dge)  
 throws [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

Creates the peer for a DragSourceContext. Always throws InvalidDndOperationException if GraphicsEnvironment.isHeadless() returns true.

**Throws:** [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### createDragGestureRecognizer

public <T extends [DragGestureRecognizer](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html)> T **createDragGestureRecognizer**([Class](http://docs.google.com/java/lang/Class.html)<T> abstractRecognizerClass,  
 [DragSource](http://docs.google.com/java/awt/dnd/DragSource.html) ds,  
 [Component](http://docs.google.com/java/awt/Component.html) c,  
 int srcActions,  
 [DragGestureListener](http://docs.google.com/java/awt/dnd/DragGestureListener.html) dgl)

Creates a concrete, platform dependent, subclass of the abstract DragGestureRecognizer class requested, and associates it with the DragSource, Component and DragGestureListener specified. subclasses should override this to provide their own implementation

**Parameters:**abstractRecognizerClass - The abstract class of the required recognizerds - The DragSourcec - The Component target for the DragGestureRecognizersrcActions - The actions permitted for the gesturedgl - The DragGestureListener **Returns:**the new object or null. Always returns null if GraphicsEnvironment.isHeadless() returns true.**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### getDesktopProperty

public final [Object](http://docs.google.com/java/lang/Object.html) **getDesktopProperty**([String](http://docs.google.com/java/lang/String.html) propertyName)

Obtains a value for the specified desktop property. A desktop property is a uniquely named value for a resource that is Toolkit global in nature. Usually it also is an abstract representation for an underlying platform dependent desktop setting. For more information on desktop properties supported by the AWT see [AWT Desktop Properties](http://docs.google.com/doc-files/DesktopProperties.html).

### setDesktopProperty

protected final void **setDesktopProperty**([String](http://docs.google.com/java/lang/String.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) newValue)

Sets the named desktop property to the specified value and fires a property change event to notify any listeners that the value has changed.

### lazilyLoadDesktopProperty

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

an opportunity to lazily evaluate desktop property values.

### initializeDesktopProperties

protected void **initializeDesktopProperties**()

initializeDesktopProperties

### addPropertyChangeListener

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

Adds the specified property change listener for the named desktop property. If pcl is null, no exception is thrown and no action is performed.

**Parameters:**name - The name of the property to listen forpcl - The property change listener**Since:** 1.2

### removePropertyChangeListener

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

Removes the specified property change listener for the named desktop property. If pcl is null, no exception is thrown and no action is performed.

**Parameters:**name - The name of the property to removepcl - The property change listener**Since:** 1.2

### getPropertyChangeListeners

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

Returns an array of all the property change listeners registered on this toolkit.

**Returns:**all of this toolkit's PropertyChangeListeners or an empty array if no property change listeners are currently registered**Since:** 1.4

### getPropertyChangeListeners

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

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

**Parameters:**propertyName - the named property **Returns:**all of the PropertyChangeListeners associated with the named property or an empty array if no such listeners have been added**Since:** 1.4

### isAlwaysOnTopSupported

public boolean **isAlwaysOnTopSupported**()

Returns whether the always-on-top mode is supported by this toolkit. To detect whether the always-on-top mode is supported for a particular Window, use [Window.isAlwaysOnTopSupported()](http://docs.google.com/java/awt/Window.html#isAlwaysOnTopSupported()).

**Returns:**true, if current toolkit supports the always-on-top mode, otherwise returns false**Since:** 1.6 **See Also:**[Window.isAlwaysOnTopSupported()](http://docs.google.com/java/awt/Window.html#isAlwaysOnTopSupported()), [Window.setAlwaysOnTop(boolean)](http://docs.google.com/java/awt/Window.html#setAlwaysOnTop(boolean))

### isModalityTypeSupported

public abstract boolean **isModalityTypeSupported**([Dialog.ModalityType](http://docs.google.com/java/awt/Dialog.ModalityType.html) modalityType)

Returns whether the given modality type is supported by this toolkit. If a dialog with unsupported modality type is created, then Dialog.ModalityType.MODELESS is used instead.

**Parameters:**modalityType - modality type to be checked for support by this toolkit **Returns:**true, if current toolkit supports given modality type, false otherwise**Since:** 1.6 **See Also:**[Dialog.ModalityType](http://docs.google.com/java/awt/Dialog.ModalityType.html), [Dialog.getModalityType()](http://docs.google.com/java/awt/Dialog.html#getModalityType()), [Dialog.setModalityType(java.awt.Dialog.ModalityType)](http://docs.google.com/java/awt/Dialog.html#setModalityType(java.awt.Dialog.ModalityType))

### isModalExclusionTypeSupported

public abstract boolean **isModalExclusionTypeSupported**([Dialog.ModalExclusionType](http://docs.google.com/java/awt/Dialog.ModalExclusionType.html) modalExclusionType)

Returns whether the given modal exclusion type is supported by this toolkit. If an unsupported modal exclusion type property is set on a window, then Dialog.ModalExclusionType.NO\_EXCLUDE is used instead.

**Parameters:**modalExclusionType - modal exclusion type to be checked for support by this toolkit **Returns:**true, if current toolkit supports given modal exclusion type, false otherwise**Since:** 1.6 **See Also:**[Dialog.ModalExclusionType](http://docs.google.com/java/awt/Dialog.ModalExclusionType.html), [Window.getModalExclusionType()](http://docs.google.com/java/awt/Window.html#getModalExclusionType()), [Window.setModalExclusionType(java.awt.Dialog.ModalExclusionType)](http://docs.google.com/java/awt/Window.html#setModalExclusionType(java.awt.Dialog.ModalExclusionType))

### addAWTEventListener

public void **addAWTEventListener**([AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html) listener,  
 long eventMask)

Adds an AWTEventListener to receive all AWTEvents dispatched system-wide that conform to the given eventMask.

First, if there is a security manager, its checkPermission method is called with an AWTPermission("listenToAllAWTEvents") permission. This may result in a SecurityException.

eventMask is a bitmask of event types to receive. It is constructed by bitwise OR-ing together the event masks defined in AWTEvent.

Note: event listener use is not recommended for normal application use, but are intended solely to support special purpose facilities including support for accessibility, event record/playback, and diagnostic tracing. If listener is null, no exception is thrown and no action is performed.

**Parameters:**listener - the event listener.eventMask - the bitmask of event types to receive **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow the operation.**Since:** 1.2 **See Also:**[removeAWTEventListener(java.awt.event.AWTEventListener)](http://docs.google.com/java/awt/Toolkit.html#removeAWTEventListener(java.awt.event.AWTEventListener)), [getAWTEventListeners()](http://docs.google.com/java/awt/Toolkit.html#getAWTEventListeners()), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTEvent](http://docs.google.com/java/awt/AWTEvent.html), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html), [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html), [AWTEventListenerProxy](http://docs.google.com/java/awt/event/AWTEventListenerProxy.html)

### removeAWTEventListener

public void **removeAWTEventListener**([AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html) listener)

Removes an AWTEventListener from receiving dispatched AWTEvents.

First, if there is a security manager, its checkPermission method is called with an AWTPermission("listenToAllAWTEvents") permission. This may result in a SecurityException.

Note: event listener use is not recommended for normal application use, but are intended solely to support special purpose facilities including support for accessibility, event record/playback, and diagnostic tracing. If listener is null, no exception is thrown and no action is performed.

**Parameters:**listener - the event listener. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow the operation.**Since:** 1.2 **See Also:**[addAWTEventListener(java.awt.event.AWTEventListener, long)](http://docs.google.com/java/awt/Toolkit.html#addAWTEventListener(java.awt.event.AWTEventListener,%20long)), [getAWTEventListeners()](http://docs.google.com/java/awt/Toolkit.html#getAWTEventListeners()), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTEvent](http://docs.google.com/java/awt/AWTEvent.html), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html), [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html), [AWTEventListenerProxy](http://docs.google.com/java/awt/event/AWTEventListenerProxy.html)

### getAWTEventListeners

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

Returns an array of all the AWTEventListeners registered on this toolkit. If there is a security manager, its checkPermission method is called with an AWTPermission("listenToAllAWTEvents") permission. This may result in a SecurityException. Listeners can be returned within AWTEventListenerProxy objects, which also contain the event mask for the given listener. Note that listener objects added multiple times appear only once in the returned array.

**Returns:**all of the AWTEventListeners or an empty array if no listeners are currently registered **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow the operation.**Since:** 1.4 **See Also:**[addAWTEventListener(java.awt.event.AWTEventListener, long)](http://docs.google.com/java/awt/Toolkit.html#addAWTEventListener(java.awt.event.AWTEventListener,%20long)), [removeAWTEventListener(java.awt.event.AWTEventListener)](http://docs.google.com/java/awt/Toolkit.html#removeAWTEventListener(java.awt.event.AWTEventListener)), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTEvent](http://docs.google.com/java/awt/AWTEvent.html), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html), [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html), [AWTEventListenerProxy](http://docs.google.com/java/awt/event/AWTEventListenerProxy.html)

### getAWTEventListeners

public [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html)[] **getAWTEventListeners**(long eventMask)

Returns an array of all the AWTEventListeners registered on this toolkit which listen to all of the event types specified in the eventMask argument. If there is a security manager, its checkPermission method is called with an AWTPermission("listenToAllAWTEvents") permission. This may result in a SecurityException. Listeners can be returned within AWTEventListenerProxy objects, which also contain the event mask for the given listener. Note that listener objects added multiple times appear only once in the returned array.

**Parameters:**eventMask - the bitmask of event types to listen for **Returns:**all of the AWTEventListeners registered on this toolkit for the specified event types, or an empty array if no such listeners are currently registered **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow the operation.**Since:** 1.4 **See Also:**[addAWTEventListener(java.awt.event.AWTEventListener, long)](http://docs.google.com/java/awt/Toolkit.html#addAWTEventListener(java.awt.event.AWTEventListener,%20long)), [removeAWTEventListener(java.awt.event.AWTEventListener)](http://docs.google.com/java/awt/Toolkit.html#removeAWTEventListener(java.awt.event.AWTEventListener)), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTEvent](http://docs.google.com/java/awt/AWTEvent.html), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html), [AWTEventListener](http://docs.google.com/java/awt/event/AWTEventListener.html), [AWTEventListenerProxy](http://docs.google.com/java/awt/event/AWTEventListenerProxy.html)

### mapInputMethodHighlight

public abstract [Map](http://docs.google.com/java/util/Map.html)<[TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html),?> **mapInputMethodHighlight**([InputMethodHighlight](http://docs.google.com/java/awt/im/InputMethodHighlight.html) highlight)  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Returns a map of visual attributes for the abstract level description of the given input method highlight, or null if no mapping is found. The style field of the input method highlight is ignored. The map returned is unmodifiable.

**Parameters:**highlight - input method highlight **Returns:**style attribute map, or null **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless returns true**Since:** 1.3 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Toolkit.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/TexturePaint.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/Transparency.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/Toolkit.html)    [**NO FRAMES**](http://docs.google.com/Toolkit.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

[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).