| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Robot.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/RenderingHints.Key.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/Scrollbar.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/Robot.html)    [**NO FRAMES**](http://docs.google.com/Robot.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **java.awt**

Class Robot

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

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

This class is used to generate native system input events for the purposes of test automation, self-running demos, and other applications where control of the mouse and keyboard is needed. The primary purpose of Robot is to facilitate automated testing of Java platform implementations.

Using the class to generate input events differs from posting events to the AWT event queue or AWT components in that the events are generated in the platform's native input queue. For example, Robot.mouseMove will actually move the mouse cursor instead of just generating mouse move events.

Note that some platforms require special privileges or extensions to access low-level input control. If the current platform configuration does not allow input control, an AWTException will be thrown when trying to construct Robot objects. For example, X-Window systems will throw the exception if the XTEST 2.2 standard extension is not supported (or not enabled) by the X server.

Applications that use Robot for purposes other than self-testing should handle these error conditions gracefully.

**Since:** 1.3

| **Constructor Summary** | |
| --- | --- |
| [**Robot**](http://docs.google.com/java/awt/Robot.html#Robot())()            Constructs a Robot object in the coordinate system of the primary screen. |
| [**Robot**](http://docs.google.com/java/awt/Robot.html#Robot(java.awt.GraphicsDevice))([GraphicsDevice](http://docs.google.com/java/awt/GraphicsDevice.html) screen)            Creates a Robot for the given screen device. |

| **Method Summary** | |
| --- | --- |
| [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) | [**createScreenCapture**](http://docs.google.com/java/awt/Robot.html#createScreenCapture(java.awt.Rectangle))([Rectangle](http://docs.google.com/java/awt/Rectangle.html) screenRect)            Creates an image containing pixels read from the screen. |
| void | [**delay**](http://docs.google.com/java/awt/Robot.html#delay(int))(int ms)            Sleeps for the specified time. |
| int | [**getAutoDelay**](http://docs.google.com/java/awt/Robot.html#getAutoDelay())()            Returns the number of milliseconds this Robot sleeps after generating an event. |
| [Color](http://docs.google.com/java/awt/Color.html) | [**getPixelColor**](http://docs.google.com/java/awt/Robot.html#getPixelColor(int,%20int))(int x, int y)            Returns the color of a pixel at the given screen coordinates. |
| boolean | [**isAutoWaitForIdle**](http://docs.google.com/java/awt/Robot.html#isAutoWaitForIdle())()            Returns whether this Robot automatically invokes waitForIdle after generating an event. |
| void | [**keyPress**](http://docs.google.com/java/awt/Robot.html#keyPress(int))(int keycode)            Presses a given key. |
| void | [**keyRelease**](http://docs.google.com/java/awt/Robot.html#keyRelease(int))(int keycode)            Releases a given key. |
| void | [**mouseMove**](http://docs.google.com/java/awt/Robot.html#mouseMove(int,%20int))(int x, int y)            Moves mouse pointer to given screen coordinates. |
| void | [**mousePress**](http://docs.google.com/java/awt/Robot.html#mousePress(int))(int buttons)            Presses one or more mouse buttons. |
| void | [**mouseRelease**](http://docs.google.com/java/awt/Robot.html#mouseRelease(int))(int buttons)            Releases one or more mouse buttons. |
| void | [**mouseWheel**](http://docs.google.com/java/awt/Robot.html#mouseWheel(int))(int wheelAmt)            Rotates the scroll wheel on wheel-equipped mice. |
| void | [**setAutoDelay**](http://docs.google.com/java/awt/Robot.html#setAutoDelay(int))(int ms)            Sets the number of milliseconds this Robot sleeps after generating an event. |
| void | [**setAutoWaitForIdle**](http://docs.google.com/java/awt/Robot.html#setAutoWaitForIdle(boolean))(boolean isOn)            Sets whether this Robot automatically invokes waitForIdle after generating an event. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/awt/Robot.html#toString())()            Returns a string representation of this Robot. |
| void | [**waitForIdle**](http://docs.google.com/java/awt/Robot.html#waitForIdle())()            Waits until all events currently on the event queue have been processed. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

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

### Robot

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

Constructs a Robot object in the coordinate system of the primary screen.

**Throws:** [AWTException](http://docs.google.com/java/awt/AWTException.html) - if the platform configuration does not allow low-level input control. This exception is always thrown when GraphicsEnvironment.isHeadless() returns true [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if createRobot permission is not granted**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless()), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html)

### Robot

public **Robot**([GraphicsDevice](http://docs.google.com/java/awt/GraphicsDevice.html) screen)  
 throws [AWTException](http://docs.google.com/java/awt/AWTException.html)

Creates a Robot for the given screen device. Coordinates passed to Robot method calls like mouseMove and createScreenCapture will be interpreted as being in the same coordinate system as the specified screen. Note that depending on the platform configuration, multiple screens may either:

* share the same coordinate system to form a combined virtual screen
* use different coordinate systems to act as independent screens

This constructor is meant for the latter case.

If screen devices are reconfigured such that the coordinate system is affected, the behavior of existing Robot objects is undefined.

**Parameters:**screen - A screen GraphicsDevice indicating the coordinate system the Robot will operate in. **Throws:** [AWTException](http://docs.google.com/java/awt/AWTException.html) - if the platform configuration does not allow low-level input control. This exception is always thrown when GraphicsEnvironment.isHeadless() returns true. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if screen is not a screen GraphicsDevice. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if createRobot permission is not granted**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless()), [GraphicsDevice](http://docs.google.com/java/awt/GraphicsDevice.html), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html)

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

### mouseMove

public void **mouseMove**(int x,  
 int y)

Moves mouse pointer to given screen coordinates.

**Parameters:**x - X positiony - Y position

### mousePress

public void **mousePress**(int buttons)

Presses one or more mouse buttons. The mouse buttons should be released using the mouseRelease method.

**Parameters:**buttons - the Button mask; a combination of one or more of these flags:

* InputEvent.BUTTON1\_MASK
* InputEvent.BUTTON2\_MASK
* InputEvent.BUTTON3\_MASK

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the button mask is not a valid combination**See Also:**[mouseRelease(int)](http://docs.google.com/java/awt/Robot.html#mouseRelease(int))

### mouseRelease

public void **mouseRelease**(int buttons)

Releases one or more mouse buttons.

**Parameters:**buttons - the Button mask; a combination of one or more of these flags:

* InputEvent.BUTTON1\_MASK
* InputEvent.BUTTON2\_MASK
* InputEvent.BUTTON3\_MASK

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the button mask is not a valid combination**See Also:**[mousePress(int)](http://docs.google.com/java/awt/Robot.html#mousePress(int))

### mouseWheel

public void **mouseWheel**(int wheelAmt)

Rotates the scroll wheel on wheel-equipped mice.

**Parameters:**wheelAmt - number of "notches" to move the mouse wheel Negative values indicate movement up/away from the user, positive values indicate movement down/towards the user.**Since:** 1.4

### keyPress

public void **keyPress**(int keycode)

Presses a given key. The key should be released using the keyRelease method.

Key codes that have more than one physical key associated with them (e.g. KeyEvent.VK\_SHIFT could mean either the left or right shift key) will map to the left key.

**Parameters:**keycode - Key to press (e.g. KeyEvent.VK\_A) **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if keycode is not a valid key**See Also:**[keyRelease(int)](http://docs.google.com/java/awt/Robot.html#keyRelease(int)), [KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html)

### keyRelease

public void **keyRelease**(int keycode)

Releases a given key.

Key codes that have more than one physical key associated with them (e.g. KeyEvent.VK\_SHIFT could mean either the left or right shift key) will map to the left key.

**Parameters:**keycode - Key to release (e.g. KeyEvent.VK\_A) **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if keycode is not a valid key**See Also:**[keyPress(int)](http://docs.google.com/java/awt/Robot.html#keyPress(int)), [KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html)

### getPixelColor

public [Color](http://docs.google.com/java/awt/Color.html) **getPixelColor**(int x,  
 int y)

Returns the color of a pixel at the given screen coordinates.

**Parameters:**x - X position of pixely - Y position of pixel **Returns:**Color of the pixel

### createScreenCapture

public [BufferedImage](http://docs.google.com/java/awt/image/BufferedImage.html) **createScreenCapture**([Rectangle](http://docs.google.com/java/awt/Rectangle.html) screenRect)

Creates an image containing pixels read from the screen. This image does not include the mouse cursor.

**Parameters:**screenRect - Rect to capture in screen coordinates **Returns:**The captured image **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if screenRect width and height are not greater than zero [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if readDisplayPixels permission is not granted**See Also:**[SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [AWTPermission](http://docs.google.com/java/awt/AWTPermission.html)

### isAutoWaitForIdle

public boolean **isAutoWaitForIdle**()

Returns whether this Robot automatically invokes waitForIdle after generating an event.

**Returns:**Whether waitForIdle is automatically called

### setAutoWaitForIdle

public void **setAutoWaitForIdle**(boolean isOn)

Sets whether this Robot automatically invokes waitForIdle after generating an event.

**Parameters:**isOn - Whether waitForIdle is automatically invoked

### getAutoDelay

public int **getAutoDelay**()

Returns the number of milliseconds this Robot sleeps after generating an event.

### setAutoDelay

public void **setAutoDelay**(int ms)

Sets the number of milliseconds this Robot sleeps after generating an event.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If ms is not between 0 and 60,000 milliseconds inclusive

### delay

public void **delay**(int ms)

Sleeps for the specified time. To catch any InterruptedExceptions that occur, Thread.sleep() may be used instead.

**Parameters:**ms - time to sleep in milliseconds **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if ms is not between 0 and 60,000 milliseconds inclusive**See Also:**[Thread.sleep(long)](http://docs.google.com/java/lang/Thread.html#sleep(long))

### waitForIdle

public void **waitForIdle**()

Waits until all events currently on the event queue have been processed.

**Throws:** [IllegalThreadStateException](http://docs.google.com/java/lang/IllegalThreadStateException.html) - if called on the AWT event dispatching thread

### toString

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

Returns a string representation of this Robot.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**the string representation.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Robot.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/RenderingHints.Key.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/Scrollbar.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/Robot.html)    [**NO FRAMES**](http://docs.google.com/Robot.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

[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).